AD-A094 610   NAVAL RESEARCH LAB WASHINGTON DC   F/G 9/2

REAL-TIME AND APPLICATIONS SOFTWARE FOR THE NEIL BROWN/W.H.O.I.--ETC(U)

DEC 80 L J ROSENBLUM

UNCLASSIFIED NRL-8444 NL

1 OF 1
AD
A094 610

END
DATE
FILMED
2-81
DTIC

LEVEL 12

NRL Report 8444

Real-Time and Applications Software for the Neil Brown/W.H.O.I. CTD Microprofiler

Lawrence J. Rosenblum

Oceanographic Computer Applications Group

December 30, 1980

NAVAL RESEARCH LABORATORY
Washington, D.C.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NRL Report 8444	2. GOVT ACCESSION NO.: AD-A094626
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): REAL-TIME AND APPLICATIONS SOFTWARE FOR THE NEIL BROWN/W.H.O.I. CTD MICROPROFILER	5. TYPE OF REPORT & PERIOD COVERED: Final report
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Lawrence J. Rosenblum	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Research Laboratory, Washington, DC 20375	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: J. O. 68056-6013
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE: December 30, 1980
	13. NUMBER OF PAGES: 46
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Oceanography
Ocean microstructure measurements
CTD microprofiler data acquisition
CTD microprofiler real-time data display
Real-time processing
Minicomputer applications

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Neil Brown/W.H.O.I. CTD Microprofiler has been extensively used by a variety of NRL researchers in at-sea experiments. This paper reports on software for both real-time data acquisition and display and for processing collected data.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

REAL-TIME AND APPLICATIONS SOFTWARE FOR THE NEIL BROWN/W.H.O.I. CTD MICROPROFILER

THE CTD MICROPROFILER

Introduction

The W.H.O.I./Brown conductivity, temperature, and depth (CTD) profiler is a device to make fine-scale measurements of these physical parameters in the ocean. The CTD was first used by the Naval Research Laboratory (NRL) in July 1977 and has been extensively used in field work in physical oceanography, chemical oceanography, acoustics, and special projects ever since. In August 1979 one of the NRL CTD devices was modified to include a fluorometer. All field work utilizing the CTDs has used on-board computer systems of the Oceanographic Computer Applications Group. This report documents software written by the author for the real-time acquisition, storage, and display of CTD data as well as application programs for processing collected data.

The CTD consists of a shipboard data terminal deck unit and an underwater unit which provides continuous sampling of the variables as it is lowered through the water. Conductivity is measured with a miniature four-electrode ceramic cell. Temperature is sensed by electronically combining the outputs of a high-speed thermistor and a platinum resistance thermometer. Pressure is sensed by a strain-gauge pressure transducer. Data from the underwater unit is transmitted in real time to the shipboard data terminal through a single electrical conductor in the core of the steel cable used to support the instrument in the water. The data are in Teletype format using frequency-shift key modulation of 5- and 10-kHz ac signals superimposed on the dc power supply to the underwater unit through the same cable. The deck unit decodes the signals, provides digital data in both parallel and serial format for computer processing, and displays conductivity, temperature, and pressure. For greater detail on the CTD see references 1 to 5.

The NRL Oceanographic Computer Applications Group uses Hewlett-Packard (HP) 1000-series computers and a variety of peripherals depending on individual requirements. For the September/October 1979 cruise aboard the USNS *Lynch* two HP21MX-E computers were used, one for taking real-time data from various physical oceanographic data collection devices and ship's sensors and the other for recording CTD and XBT data. The CTD system produced real-time plots on a HP2647 terminal and used an HP2631G line printer for producing hard copy of these plots. Minor program modifications are necessary for other peripherals. Figure 1, taken aboard the USNS *Hayes* in 1977, shows a typical shipboard computer system.

Interfacing

The CTD offers both serial and parallel data ports for use in interfacing. In the CTD data flow each byte (8 bits) is preceded by one bit containing a logic "zero" and terminated by two bits, each of which contains a logic "one." When a digitized channel requires two bytes to convey magnitude, the first byte is the least significant byte. The original interface, developed by the Oceanographic Computer Applications Group in 1977, performed a sample-and-hold operation on an entire scan of data holding the three 16-bit words containing pressure, temperature, and conductivity. These three words are then sent to the computer on demand using three computer microcircuit interface boards, one for each parameter. A driver developed by the Oceanographic Computer Applications Group for doing sampling of successive I/O channels on HP1000-series computers was employed to control the I/O.

Figure 1 — Computer system aboard USNS *Hayes* in 1977

The original approach had some disadvantages. While syncronization problems have been found to be rare, the "sync" and utility frames were unavailable. Also, the requirement for three successive channels was inconvenient and occasionally necessitated the use of an I/O extender to provide additional computer-interface channels. The extender requires additional rack space and creates the potential for hardware failure aboard ship. Accordingly, the CTD system required revision to work with a CTD microprofiler modified to include fluorometer measurements. The data were still taken at the deck unit parallel output port; however, the interrupt strobes were "or"ed together and data was sent to the computer in word serial mode with no indication of which word (or byte) of data was being sent. Thus, the software had to determine where the sync frame was in order to start data acquisition and to assure that the data flow remained in sync. The data were sent to the computer through an I/O board using the same driver as before. This method brings in all the collected data bytes, but greater programming care was required since the internal timing of the CTD within a scan had to be considered in the programming to assure that sufficient time was available between words to perform all CTD operations without data loss.

The fluorometer CTD system has the following data format:

Byte 1: Frame sync. Alternate between 11110000 and 00001111
Byte 2: Pressure (LSBs)

Byte 3: Pressure (MSBs)
Byte 4: Temperature (LSBs)
Byte 5: Temperature (MSBs)
Byte 6: Conductivity (LSBs)
Byte 7: Conductivity (MSBs)
Byte 8: Utility. Contains pressure and temperature signs.
Byte 9: Fluorometer "Signal." All "ones" correspond to full scale input.
Byte 10: Fluorometer "Range." All "ones" correspond to full scale input.

As currently interfaced, pressure, temperature, and conductivity are packed into words by the interface unit, with the remaining parameters placed in the lower half of full words.

This report deals with the CTD software for the fluorometer CTD. However, both program sets are quite similar and differences are likely to be transparent to a user.

REAL-TIME COLLECTION AND DISPLAY SOFTWARE

The Real-Time Program Set

The real-time data collection and display programs for the fluorometer CTD unit are described in this chapter.

There are nine real-time programs used. These are:

CTDRT — Obtains data and schedules other programs.
CTDIN — Initializes parameters prior to data collection and writes an initialization record on magnetic tape.
CTDMT — Writes data records on magnetic tape.
CTDP1 — Produces real-time plots of salinity and temperature vs pressure.
CTDP2 — Produces real-time plots of salinity, temperature, and pressure vs time.
CTDCN — Changes certain control functions while data are being taken.
CTDTR — Writes termination file.
CTDMS — Writes message to terminal for CTDRT.
CTDFL — Sets fluorometer On/Off switch.

System Requirements

The programs are designed to operate under RTE-II, RTE-III, or RTE-IV (Real-Time Executive operating systems) on HP1000-series computers with 64K words of memory. A background system common area of 2000_8 is required. The collection program CTDRT is loaded as a foreground program using reverse (background) common. The remaining programs are loaded as background programs using background common.

The programs listed in this section utilize the HP2647 terminal and HP2631G printer. The HP2631G is also used to produce hard copy of terminal plots. The programs require minor revisions if other peripherals are used since the means for producing hard copy of plots is device dependent and also the screen size of the HP2647 requires a minor scaling modification of the plot.

Data File Structure

One CTD lowering (or towing) results in a data tape file with the following format:

| INITIALIZATION RECORD | EOF | DATA RECORDS | EOF | TERM RECORD | EOF | EOF |

Prior to the next event, the program F2EOF is run to position the tape between the double end-of-file marks and the next initialization record is written over the last EOF. Hence a tape with several CTD events will have a double EOF only after the last event. A sequence of three events would appear as

	E		E		E		E		E		E		E		E		E	E
Init.	O	Data	O	Term.	O	Init.	O	Data	O	Term.	O	Init.	O	Data	O	Term.	O	O
Record	F		F	Record	F	Record	F		F	Record	F	Record	F		F	Record	F	F

The initialization record is a 47-word header record containing identification parameters for the event:

Words	1-6:	12 Character Identifying ASCII String
Word	7:	Station Number
Word	8:	Year
Word	9:	Month
Word	10:	Day
Word	11:	Lowering Number
Word	12:	Instrument Number
Words	13-24:	Conversion Constants: C1, . . . , C6
Words	25-30:	Pressure Axis Constants
Words	31-36:	Salinity Axis Constants
Words	37-42:	Temperature Axis Constants
Words	43-44:	Latitude
Words	45-46:	Longitude
Word	47:	Zero

The initialization record is followed by an EOF and then the CTD data. Each data record contains 456 words (approximately 2 seconds of data) consisting of 8 words of header information followed by 448 data words:

Word	1:	Record Number
Word	2:	Time (Hours = 24 × Days + Hours)
Word	3:	Time (Seconds = 60 × Minute + Seconds)
Word	4:	Time (Tens of milliseconds)
Word	5:	Special Segment ID Marker
Word	6:	Fluorometer on/off switch
Word	7:	0
Word	8:	0
Words	9-456:	Digital Data

The last data record is followed by an EOF and an 8-word termination record:

Word	1:	−1
Word	2:	Time Hours
Word	3:	Time Seconds
Word	4:	Termination Method Indicator
Words	5-8:	0

Finally, a double EOF is written on the tape.

Data Acquisition and Display

The data-acquisition programs are started by running the program CTDRT (a full description of questions and answers is given in the next chapter). CTDRT then schedules programs CTDMS for execution. CTDMS asks the operator whether the operation will be a CTD lowering or tow and if temperature or potential temperature is to be a plotting parameter. For a lowering, temperature (or potential temperature) and salinity are both plotted against pressure, while in the case of a tow, temperature (or potential temperature), salinity, and pressure are plotted as a function of time. Note that pressure in decibars is very nearly identical to depth in meters.

Now that the plot type has been defined, program CTDRT schedules programs CTDIN for execution to obtain identifying parameters and plot scale parameters. These parameters are written to magnetic tape in a 47-word header record for the CTD cast. The calibration constants are used to convert the CTD measurement into pressure, temperature, and salinity. If M_1, M_2, M_3 are the real values of the measurements obtained from the CTD unit for conductivity pressure and temperature, then

$$\begin{aligned} \text{Pressure} &= C1 * (M_2 - C2) \\ \text{Temperature} &= C3 + (C4 * M_3) \\ \text{Conductivity} &= C5 * M_1 * (1 + C6 * \text{Temperature}) \end{aligned}$$

$C1$ is the only constant with which the user need be concerned; it depends upon the pressure sensor used within the CTD. The axes parameters determine the parameter limits for the real-time CTD display. Points outside the user-defined axes limits will not appear on the real-time plot, so the user should exercise care in assuring that these limits span the data without being so wide that the display loses meaning. Note, however, that these limits affect only the real-time display and not the collection of the data or the process of writing the data records onto magnetic tape. Thus, if the date is outside the user-defined limits, the ability to view the data in real-time is lost but the collection process will not be affected.

Once the initialization parameters have been obtained, the program CTDIN displays the values and asks if the values are correct. If a mistake has been made while inputting the parameters, the user answers NO and the questions are repeated. If they are correct, CTDIN writes the initialization record and an end-of-file (EOF) onto magnetic tape and terminates, which allows the scheduling program CTDRT to resume.

Depending upon whether a lowering or towing real-time display has been requested, CTDRT next schedules either program CTDP1 or CTDP2. In either case the appropriate axes are drawn on a plotting terminal, with the axes parameters obtained by CTDIN. When these axes have been drawn, CTDP1 (or CTDP2) suspends execution with the point of suspension being retained. This enables plotting to occur without the screen being erased each time a pair of points are to be drawn. Program CTDRT now has program CTDMS print the message "PAUSE—TYPE: 'GO, CTDRT' FOR DATA ACQUISITION" on the system console and then suspends. This completes the initialization phase. The operator then types "GO, CTDRT" on the system console to initiate data collection.

Figures 2 and 3 were made aboard the USNS *Lynch* in October 1977 near Bermuda and show typical plots for the lowering mode. For Fig. 2 the CTD was lowered near the surface while in Fig. 3 the device "yo-yoed" near the bottom. Figures 4 and 5 were made during the return run from Bermuda to Newport, Rhode Island, and show the real-time display for the towing mode. Here pressure, temperature, and salinity are displayed as a function of time.

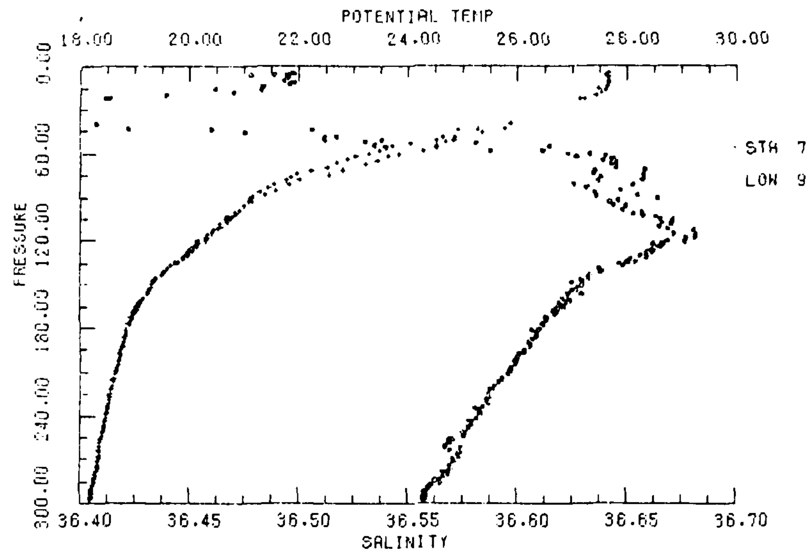


Figure 2 — Real-time CTD plot for near-surface lowering

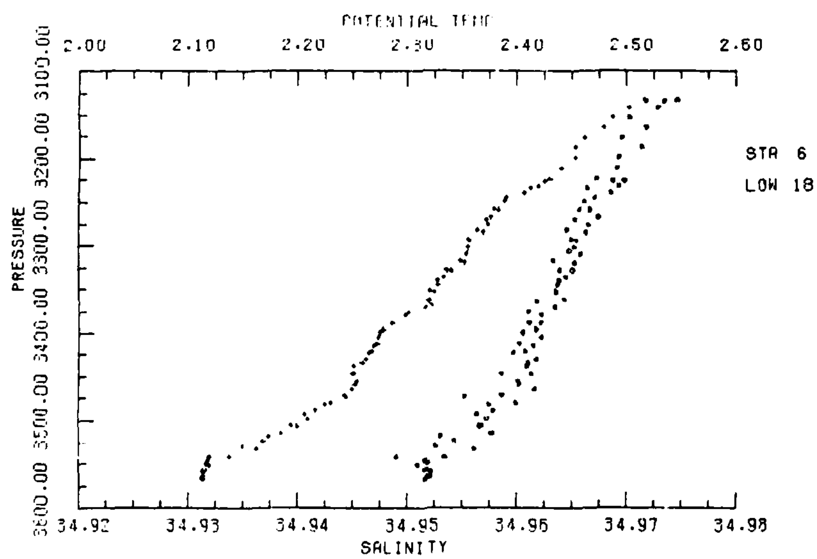


Figure 3 — Real-time CTD—"yo-yo" in vicinity of bottom

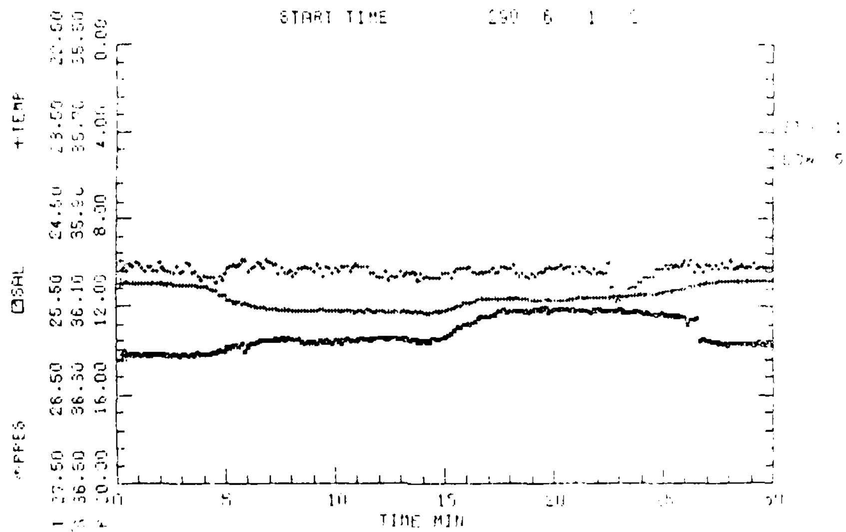


Figure 4 — Real time CTD plot—"tow" mode

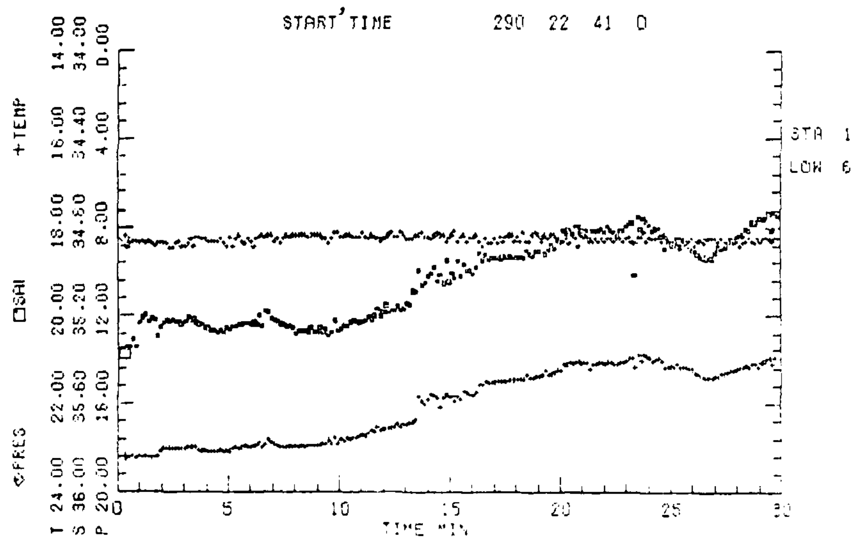


Figure 5 — Real time CTD plot—"tow" mode

The basic programming technique used for data acquisition and processing is double buffering. The data are acquired in (approximately) 2-second buffers and the processing of the previous buffer must be accomplished within the 2 seconds allotted for obtaining the new data. Since the CTD scan rate is 31.25 hertz, one scan takes 32 milliseconds. As previously discussed, the CTD interface sends the data words in a serial format. The CTD interface, in one scan, sends the computer:

Word 1: Sync Frame
Word 2: Pressure
Word 3: Temperature
Word 4: Conductivity
Word 5: Utility
Word 6: Fluorometer "Signal"
Word 7: Fluorometer "Range"

where those words which contain only one byte of information have the data in the LSBs.

Returning to the data processing flow, program CTDRT compares words against the CTD sync frame values until the sync frame is recognized. It next determines which of the two buffers is to be filled. Once this is done, program CTDRT schedules program CTDMT and requests 448 words of data from the CTD interface. Thus, while the foreground program CTDRT obtains the next 2 seconds of data, the background programs (CTDMT and either CTDP1 or CTDP2) processes the previous 2 seconds of data. CTDMT keeps track of the buffer times and fills in the appropriate entries in the header—the first 8 words of the buffer as given above. The record is then dumped onto magnetic tape, and every fourth buffer the plotting program CTDP1 (or CTDP2) is scheduled to plot a point and print out on the line printer and system console the values of temperature, pressure, and salinity. These values are printed even if a point is not plotted due to one of the values being off the plot scale. When program CTDP1 (or CTDP2) is scheduled for execution, it takes the first three data points in the buffer and uses the constants $C1, C2, \ldots, C6$ to obtain pressure, temperature, and conductivity. Salinity is computed using an algorithm of Fofonoff. These points are then scaled for plotting and, if within the user defined plot axes limits, they are plotted on the real-time plot display. The values are printed out as mentioned above. If the CTD is being operated in the tow mode each plot will contain 1/2 hour of data. At the end of the half-hour program CTDP2 will automatically copy the plot onto a hard-copy unit (the 2631G terminal, a Tektronix hardcopier, etc., depending upon the system in use). The axes are then redrawn and reannotated with the same value as previously used and the plot continues.

The remaining programs are used for various control functions. Program CTDFL sets the fluorometer on/off switch (Word 6 of the buffer). it should be noted that this switch is for convenience in locating fluorometer data, but is not used to alter the data; hence, operator failure to properly set the switch will not affect the data tape. The program CTDCN is used for several purposes. It can be used to put a marker in word 5 of the data record in order to identify special segments of data for easy future location. Alternately, program CTD is used to put the CTD programs in "idle" mode and to terminate data collection. Occasionally, in the course of a CTD lowering, events such as cable snags will necessitate holding the CTD at a fixed depth for a certain period, and it has been found useful to have the capability of suspending data collection while such problems are being corrected. Accordingly, one mode of operation of program CTDCN will put all programs into a suspend state in such a way that the operator can resume data collection by typing a command on the system console. CTDCN is also used to terminate data collection. Full details of how to use program CTDCN are given the next section.

When program CTDCN is executed in either its termination or its abort mode, it assures that the real-time programs are properly terminated, writes an EOF onto the magnetic tape, and then schedules program CTDTR which writes an 8-word termination file and two EOFs. This completes the data collection process for one event. For further collection using the same magnetic tape, the tape is positioned between the EOF's and the process repeated.

Real-Time Programs—FORTRAN Source Listings

```
FTN4,L
      PROGRAM CTDRT(10,15)
C
C        REAL TIME PROGRAMS FOR COLLECTION OF CTD DATA WITH FLOROMETER
C        CAPABILITY.  DOUBLE BUFFERED DATA TAKEN EVERY TWO SECONDS.
C
C        REAL TIME PROGRAMS USED ARE:
C        CTDRT - SCHEDULES OTHER PROGRAMS AND CONTROL PARAMETERS
C        CTDIN- INITIALIZES PARAMETERS AND WRITES INIT FILE ON MT
C        CTDMT - WRITES DATA BUFFERS TO MT
C        CTDP1- REAL TIME PLOT OF SAL, TEMP VS. PRESSURE
C        CTDP2- REAL TIME PLOT OF SAL, TEMP, PRESS VS. TIME
C        CTDTR- WRITES TERMINATION FILE
C        CTDCN- USES COMMON PARAM ISTOP TO CONTROL FUNCTIONS
C        CTDMS- USED TO WRITE MESSAGES FOR CTDRT
C        CTDFL- SETS FLOROMETER ON/OFF SWITCH
C        OFF LINE PROGRAMS USED ARE:
C        PLTBK - USED FOR POST PROCESSING OF DATA
C        F2EOF - POSITIONS MT BETWEEN DBL EOF FOR NEW RT DATA
C        MTPOS - FINDS STA & LOW NUMBERS AND POSITIONS MT AT DATA
C
C        DATA FILE FORMAT
C        WORD 1      RECORD NUMBER
C        WORD 2      TIME(HOURS)
C        WORD 3      TIME(SECONDS)
C        WORD 4      TIME(10XMSEC)
C        WORD 5      ID MARKER FOR SPECIAL SEGMENTS - SEE CNTL
C        WORD 6      FLOROMETER ON/OFF 0=OFF 1=ON
C        WORDS 7-8   0
C        WORDS 9-456 DIGITAL DATA
C
C        SEE CTDIN FOR 47 WORD INITIALIZATION FILE AND CTDTR FOR 8 WORD
C        TERMINATION FILE
C
C        SCIENTIFIC SUBROUTINES USED ARE TAKEN FROM WOODS HOLE CTD
C        PROGRAM AND ARE LISTED WITH THE USER PROGRAM
C
C        PROGRAMMER: LARRY ROSENBLUM
C        VERSION: SEPT 20,1979
C
      DIMENSION   NAME(3),NAME1(3),NAME2(3),NAME3(3),ITIME(5)
      DIMENSION NAME4(3),ISYNC(7),NAME5(3),IPRAM(5)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      DATA NAME/2HCT,2HDM,2HT /
      DATA NAME1/2HCT,2HDI,2HN /
      DATA NAME2/2HCT,2HDT,2HR /
      DATA NAME3/2HCT,2HDP,2H1 /
      DATA NAME4/2HCT,2HDP,2H2 /
```

```
      DATA NAME5/2HCT,2HDM,2HS /
      LUCTD=10
      MSK1=177400B
      MSK2=   377B
C       CHOOSE TYPE OF PLOT WANTED
      CALL EXEC(9,NAME5,1)
      CALL RMPAR(IPRAM)
      IANS=IPRAM(2)
      IANS1=IPRAM(3)
      ISTOP=-1
      IREC=1
      IFLOR=0
C       SCHEDULE INIT AND PLT1 OR PLT2
      CALL EXEC(9,NAME1,IANS)
      IF(IANS.EQ.1)CALL EXEC(10,NAME3,IANS1)
      IF(IANS.EQ.2)CALL EXEC(10,NAME4,IANS1)
C       WAIT UNTIL AXES HAVE BEEN DRAWN & EXPT READY TO COLLECT DATA
      CALL EXEC(9,NAME5,2)
      PAUSE
C       GET INTO SYNC WITH THE SYNC FRAME
2     CALL EXEC(1,LUCTD,ISYNC,1)
      ITST=IAND(ISYNC(1),MSK1)
      IF(ITST.NE.170000B)GO TO 2
      CALL EXEC(1,LUCTD,ISYNC,6)
C       DETERMINE WHICH BUFFER IS TO BE FILLED
12    KPAR=IREC-2*(IREC/2)
      IF(KPAR.EQ.0)GO TO 40
C       ODD RECORD NUMBER
      IBUF1(1)=IREC
      IBUF1(6)=IFLOR
C       SCHEDULE MAGTP
      IF(IREC.NE.1)CALL EXEC(10,NAME,IANS,IANS1)
C       AQUIRE DATA
60    CALL EXEC(1,LUCTD,IBUF1(9),448)
      IREC=IREC+1
      IF (ISTOP.GE.0) GO TO 99
      IF(IREC.LT.4)GO TO 12
      ITST1=IAND(MSK1,IBUF2(9))
      IF(ITST1.EQ.170000B)GO TO 12
      IF(ITST1.EQ.7400B)GO TO 12
      ITST1=IAND(MSK2,IBUF2(9))
      IF(ITST1.EQ.360B)GO TO 12
      IF(ITST1.EQ.17B)GO TO 12
      GO TO 2
C       EVEN RECORD NUMBER
40    IBUF2(1)=IREC
      IBUF2(6)=IFLOR
C       SCHEDULE MAGTP
      CALL EXEC(10,NAME,IANS,IANS1)
C       AQUIRE DATA
      CALL EXEC(1,LUCTD,IBUF2(9),448)
      IREC=IREC+1
      IF(ISTOP.GE.0)  GO TO 99
      IF(IREC.LT.4)GO TO 12
```

```
      ITST1=IAND(MSK1,IBUF1(9))
      IF(ITST1.EQ.170000B)GO TO 12
      IF(ITST1.EQ.007400B)GO TO 12
      ITST1=IAND(MSK2,IBUF1(9))
      IF(ITST1.EQ.360B)GO TO 12
      IF(ITST1.EQ.17B)GO TO 12
      GO TO 2
C       END OR IDLE
99    IF(ISTOP.NE.2) GO TO 98
      CALL EXEC(9,NAME5,3)
      PAUSE
      ISTOP=-1
      GO TO  2
98    CALL EXEC(9,NAME2,IANS)
      END
      END$
```

```
FTN4,L
      PROGRAM CTDIN
C
C       PROG IS CALLED (WITH WAIT) BY CTDRT TO INITIALIZE PARAMS.
C       PARAMETER NAMES AND INITIALIZATION FILE WERE MADE TO
C       CORESPOND TO THE WOODS HOLE PROGRAM.  SOME PARAMETERS
C       ARE NOT USED IN PROGRAM.  47 WORD FILE IS WRITTEN TO MT
C       ON COMPLETION.
C       INSTR=INSTRUMENT NUMBER
C       C1-C6 ARE CONSTS USED TO CONVERT MEAS. TO PRES,TEMP, & COND
C       AXES PARAMS ARE AX LENGTH,INIT. VALUE, & INCR/INCH.  AXES
C       LENGHTH IS 6" FOR X AXIS AND 5" FOR Y AXIS.
C
C       FILE FORMAT:
C         WORDS 1-6         12 CHARACTER INPUT I. D. STRING
C         WORD 7            STATION NUMBER
C         WORDS 8-10        DATE: YEAR, MONTH, DAY
C         WORDS 11-12       LOWERING #, INSTRUMENT #
C         WORDS 13-24       CALIBRATION CONSTS (REAL VALUED)
C         WORDS 25-30       PRESSURE AXIS CONSTS (REAL VALUED)
C         WORDS 31-36       TEMPERATURE AXIS CONSTS (REAL VALUED)
C         WORDS 37-42       SALINITY AXIS CONSTS (REAL VALUED)
C         WORDS 43-46       LATITUDE & LONGITUDE (REAL VALUED)
C         WORD 47           0
C
C       PROGRAMMER: LARRY ROSENBLUM
C
      DIMENSION IBUF5(47),IPRAM(5)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      EQUIVALENCE (XC1,IBUF5(13)),(XC2,IBUF5(15))
```

```
      EQUIVALENCE (XC3,IBUF5(17)),(XC4,IBUF5(19)),(XC5,IBUF5(21))
      EQUIVALENCE (XC6,IBUF5(23)),(XPAXL,IBUF5(25)),(XPIV,IBUF5(27))
      EQUIVALENCE (XPINCR,IBUF5(29)),(XTAXL,IBUF5(31)),(XTIV,IBUF5(33))
      EQUIVALENCE (XTINCR,IBUF5(35)),(XSAXL,IBUF5(37)),(XSIV,IBUF5(39))
      EQUIVALENCE (XSINCR,IBUF5(41)),(XLAT,IBUF5(43)),(XLON,IBUF5(45))
      CALL RMPAR(IPRAM)
C       DEFAULT VALUES
    5 INSTR=1
      C1=0.1
      C2=0.0
      C3=0.0
      C4=.0005
      C5=0.0010
      C6=0.0
      PAXL=5.
      PIV=0.
      PINCR=1000.
      TAXL=6.
      IF(IPRAM(1).EQ.2)TAXL=5.
      TIV=1.
      TINCR=3.0
      IF(IPRAM(1).EQ.2)TINCR=4.0
      SAXL=6.
      IF(IPRAM(1).EQ.2)SAXL=5.
      SIV=32.
      IF(IPRAM(1).EQ.2)SIV=33.
      SINCR=1.0
      IBLNK=20040B
      DO 37 I=1,6
      ISHIP(I)=IBLNK
   37 CONTINUE
      WRITE(1,10)
   10 FORMAT("PROGRAM CTDIN RUNNING")
C       READ IN PARAMETER VALUES
      WRITE(1,12)
   12 FORMAT("ENTER@ SHIP + CRUISE # (12 CHARACTERS)")
      READ(1,13)(ISHIP(I),I=1,6)
   13 FORMAT(6A2)
      WRITE(1,14)
   14 FORMAT("ENTER DATE(YR,MO,DAY)")
      READ(1,*)IYR,IMO,IDAY
      WRITE(1,16)
   16 FORMAT("ENTER STATION NUMBER AND LOWERING NUMBER")
      READ(1,*)IST,LOW
      WRITE(1,17)
   17 FORMAT("ENTER LATITUDE: DEG, MIN & FRACT. OF MIN XXXX.,XX.XXX")
      READ(1,*)XLAT1,XLAT2
      WRITE(1,19)
   19 FORMAT("ENTER LONGITUDE:DEG, MIN & FRACT. OF MIN XXXX.,XX.XXX")
      READ(1,*)XLON1,XLON2
      XLAT=60.0*XLAT1+XLAT2
      XLON=60.0*XLON1+XLON2
      WRITE(1,18)
   18 FORMAT("USE DEFAULT VALUES, 1=YES 2=NO")
      READ(1,*)KANS
      IF(KANS.EQ.1)GO TO 99
      WRITE(1,20)
   20 FORMAT("DEFAULT VALUES GIVEN IN PARENTHS")
      WRITE(1,22)
```

```
 22   FORMAT("INSTRUMENT # (I)")
      READ(1,*)INSTR
      WRITE(1,26)
 26   FORMAT("CAL CONST PRESSURE C1,C2 (0.1,0.0)")
      READ(1,*)C1,C2
      WRITE(1,28)
 28   FORMAT("CAL CONST TEMP C3,C4 (0.0,0.0005)")
      READ(1,*)C3,C4
      WRITE(1,30)
 30   FORMAT("CAL CONST COND C5,C6 (0.001,0.0)")
      READ(1,*)C5,C6
      WRITE(1,31)
 31   FORMAT("FOR EACH AXIS THE AXIS LENGTH IS GIVEN IN PARENTHESIS")
      WRITE(1,33)
 33   FORMAT("ENTER THE INITIAL VALUE AND THE INCREMENT/INCH")
      WRITE(1,32)PAXL
 32   FORMAT("PRESSURE(AX LNTH=",I2,")ENTER: INIT VALUE, INCR INCH")
      READ(1,*)PIV,PINCR
      WRITE(1,34)TAXL
 34   FORMAT("TEMP(AX LNTH=",I2,")ENTER INIT VALUE, INCR/INCH")
      READ(1,*)TIV,TINCR
      WRITE(1,36)SAXL
 36   FORMAT("SAL(AX LNTH=",I2,")ENTER INIT VALUE, INCR.INCH")
      READ(1,*)SIV,SINCR
C       WRITE OUT VALUES
 99   DO 98 I=1,6,5
      WRITE(I,60)(ISHIP(J),J=1,6)
 60   FORMAT(1X,6A2)
      WRITE(I,62) IYR,IMO,IDAY,IST,LOW
 62   FORMAT(1X, "DATE ",3I4," STA ",I4," LOW ",I4)
      WRITE(I,64)INSTR
 64   FORMAT(1X,"INSTR ",I2)
      WRITE(I,66)C1,C2,C3,C4,C5,C6
 66   FORMAT(1X,"CAL CONSTS ",6F9.4)
      WRITE(I,68)PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
 68   FORMAT(1X,"AXES CONSTS ",9F6.2)
      WRITE(I,70)XLAT1,XLAT2,XLON1,XLON2
 70   FORMAT("LAT:",F4.0,"DEG",F6.2,"MIN",4X,"LON:",F4.0,"DEG",
     1       F6.2,"MIN")
 98   CONTINUE
      WRITE(1,105)
 105  FORMAT("ARE INPUT PARAMS O.K.: 1=YES, 2=NO, 3=YES - NO MT FILE")
      READ(1,*)IANS
      IF(IANS.EQ.2)GO TO 5
C       WRITE FILE OUT TO TAPE
      IF(IANS.EQ.3) GO TO 199
      DO 110 I=1,6
      IBUF5(I)=ISHIP(I)
 110  CONTINUE
      IBUF5(7)=IST
      IBUF5(8)=IYR
      IBUF5(9)=IMO
      IBUF5(10)=IDAY
      IBUF5(11)=LOW
      IBUF5(12)=INSTR
      IBUF5(47)=0
      XC1=C1
      XC2=C2
      XC3=C3
```

```
      XC4=C4
      XC5=C5
      XC6=C6
      XPAXL=PAXL
      XPIV=PIV
      XPINCR=PINCR
      XTAXL=TAXL
      XTIV=TIV
      XTINCR=TINCR
      XSAXL=SAXL
      XSIV=SIV
      XSINCR=SINCR
C       WRITE RECORD AND EOF
      CALL EXEC(2,8,IBUF5,47)
      CALL EXEC(3,8+100B)
  199 END
      END$
```

```
FTN4,L
      PROGRAM CTDMT(3,90)
C
C       PROGRAM TO WRITE OUT DOUBLE BUFFERED DATA COLLECTED BY
C       CTDRT TO MAG. TAPE.  ALSO SCHEDULES RT PLOTTING PROG AND
C       GETS TIME.
C
C       PROGRAMMER: LARRY ROSENBLUM
C       VERSION SEPT 22 1979
C
C       NOTE: 1ST RECORD MAY HAVE BAD TIME
C
      DIMENSION ITIME(5),IPRAM(5),NAME3(3),NAME4(3)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      DATA NAME3/2HCT,2HDP,2H1 /
      DATA NAME4/2HCT,2HDP,2H2 /
5     CALL RMPAR(IPRAM)
      IANS =IPRAM(1)
      IANS1=IPRAM(2)
      KPAR=IREC-2*(IREC/2)
C       GET CURRENT TIME
      CALL EXEC(11,ITIME)
      IHR=24*ITIME(5)+ITIME(4)
      ISEC=60*ITIME(3)+ITIME(2)
      IMSC=ITIME(1)
C       SCHED. PLOTTING PROG EVERY 4TH TIME
      TEST=IREC-4*(IREC/4)
```

```
      IF(TEST.NE.3)GO TO 15
      IF(IANS.EQ.1)CALL EXEC(10,NAME3,IANS1)
      IF(IANS.EQ.2)CALL EXEC(10,NAME4,IANS1)
15    IF (KPAR.EQ.0) GO TO 10
      IBUF2(2)=IHR1
      IBUF2(3)=ISEC1
      IBUF2(4)=IMSC1
      CALL EXEC(2,8,IBUF2,456)
      GO TO 20
10    IBUF1(2)=IHR1
      IBUF1(3)=ISEC1
      IBUF1(4)=IMSC1
      CALL EXEC(2,8,IBUF1,456)
C       SAVE OLD TIME
20    IHR1=IHR
      ISEC1=ISEC
      IMSC1=IMSC
      IF(ISTOP.EQ.1.OR.ISTOP.EQ.3)GO TO 99
      CALL EXEC(6,0,1)
      GO TO 5
99    END
      END$
```

```
FTN4,L
      PROGRAM CTDP1
C
C       REAL TIME PLOTTING PROGRAM FOR CTD.
C       USES VALUE OF IREC FROM COMMON TO PLOT ONE POINT EVERY 8TH
C       BUFFER.  USES VALUE OF ISTOP TO END OR REINITIALIZE.
C       RELOCATABLES: %LRSAL, %R, %POTMP
C
C       PROGRAMMER: LARRY ROSENBLUM
C       VERSION: SEPT 18, 1979
C
      DIMENSION DAT(3),NAMES(4),NAMET(6),NAMEP(4),NASTA(2),NALOW(2)
      DIMENSION IPRAM(5),NAPOT(7)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      DATA NAMEP/2HPR,2HES,2HSU,2HRE/
      DATA NAMES/2HSA,2HLI,2HNI,2HTY/
      DATA NAMET/2HTE,2HMP,2HER,2HAT,2HUR,2HE /
      DATA NAPOT/2HPO,2HTE,2HNT,2HIA,2HL ,2HTE,2HMP/
      DATA NASTA/2HST,2HA /
      DATA NALOW/2HLO,2HW /
      CALL RMPAR(IPRAM)
      IANS1=IPRAM(1)
      CALL EXEC(22,0)
```

```
 7    CALL PLTLU(7)
C       DRAW AXES, LABEL, AND ANNOTATE
      NSTIC=INT(SAXL)+1
      NPTIC=INT(PAXL)+1
      NTTIC=INT(TAXL)+1
C       NEXT LINE TO FUDGE FOR 2647 TERMINAL
      PAXL=0.8*PAXL
      CALL AX(0.6,0.4,SAXL,NSTIC,3,0.0)
      CALL AX(0.6,PAXL+0.4,PAXL,NPTIC,3,-90.)
      CALL AX(.6+TAXL,.4+PAXL,TAXL,NTTIC,3,-180.)
      CALL LABEL(0.6,0.2,.1,SAXL,NSTIC,SIV,SINCR,2,0.0)
      CALL SYMB(3.2,0.0,.1,NAMES,0.0,8)
      CALL LABEL(0.3,PAXL+0.4,.1,-PAXL,NPTIC,PIV,PINCR,2,+90.)
      CALL SYMB(0.1,2.4,.1,NAMEP,90.0,8)
      CALL LABEL(.6,.6+PAXL,.1,TAXL,NTTIC,TIV,TINCR,2,0.)
      IF(IANS1.EQ.1)CALL SYMB(3.00,.8+PAXL,.1,NAMET,0.0,12)
      IF(IANS1.EQ.2)CALL SYMB(3.00,.8+PAXL,.1,NAPOT,0.0,14)
      CALL SYMB(.7+SAXL,-.4+PAXL,.1,NASTA,0.0,4)
      CALL SYMB(.7+SAXL,-.7+PAXL,.1,NALOW,0.0,4)
      RIST=R(IST)
      RLOW=R(LOW)
      CALL LABEL(1.2+SAXL,-.4+PAXL,.1,0.,0.,RIST,0.0,-1,0.0)
      CALL LABEL(1.2+SAXL,-.7+PAXL,.1,0.,0.,RLOW,0.0,-1,0.0)
      CALL PLOT(0.6,0.4,-3)
      JREC=3
      KOUNT=0
C       RESET PAXL TO ORIG. VALUE FOR COMPUTATION (2647 CORRECT)
      PAXL=1.25*PAXL
C       SUSPEND
 30    CALL EXEC(6,0,1)
C       CHECK FOR END OR PLOT REINITIALIZATION
      IF(ISTOP.EQ.1.OR.ISTOP.EQ.3) GO TO 99
      IF(ISTOP.EQ.-2)GO TO 98
      RE1=R(IBUF2(12))
      RE2=R(IBUF2(10))
      RE3=R(IBUF2(11))
      DAT(2)=C1*(RE2-C2)
      DAT(3)=C3+(C4*RE3)
      DAT(1)=C5*RE1*(1+(C6*DAT(2)))
C       GET TEMPERATURE SIGN
C        **** CTD HAS PROBLEM WITH SIGN BYTE ****
C     ISGN1=IAND(IBUF2(13),2B)
C     IF(ISGN1.EQ.2)DAT( 3)=-DAT( 3)
C       COND=DAT1, PRESS=DAT2, TEMP=DAT3
C       CALL SALIN CONVERSION SUBROUTINE.  REPLACE COND WITH SALIN IN DAT1
      CALL LRSAL(DAT(1),DAT(2),DAT(3),S)
      IF(IANS1.EQ.2)WRITE(1,779)DAT(3)
 779  FORMAT("TEMP=",F10.4)
      DAT(1)=S
C            DAT1=SAL DAT2=PRESS DAT3=TEMP
C       IF REQUIRED, CONVERT TEMP INTO POTENTIAL TEMP
      IF(IANS1.EQ.2)CALL POTMP(DAT(2),DAT(3),DAT(1),PTMP)
      IF(IANS1.EQ.2)DAT(3)=PTMP
 21   CALL PLT(DAT(1),DAT(3),DAT(2))
      GO TO 30
 98   ISTOP=-1
      GO TO 7
 99   END
      SUBROUTINE PLT(S,T,P)
```

```
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      WRITE(1,40)S,T,P
      WRITE(6,40)S,T,P
40    FORMAT(3F10.4)
      SMAX=SIV+(SINCR*SAXL)
      TMAX=TIV+(TINCR*TAXL)
      PMAX=PIV+(PINCR*PAXL)
      IF(S.LT.SIV.OR.S.GT.SMAX)RETURN
      IF(T.LT.TIV.OR.T.GT.TMAX)RETURN
      IF(P.LT.PIV.OR.P.GT.PMAX)RETURN
      SSCAL=SAXL/(SMAX-SIV)
      SPLT=SSCAL*(S-SIV)
      TSCAL=TAXL/(TMAX-TIV)
      TPLT=TSCAL*(T-TIV)
      PSCAL=PAXL/(PMAX-PIV)
      PPLT=PSCAL*(PMAX-P)
      NUMB1=1
      NUMB2=3
C        ADJUST Y SCALE FOR 2647 TERMINAL
      PPLT=0.8*PPLT
      CALL SYMB(SPLT,PPLT,.03,NUMB1,0.0,-1)
      CALL SYMB(TPLT,PPLT,.03,NUMB2,0.0,-1)
      RETURN
      END
      END$
```

```
FTN4,L
      PROGRAM CTDP2
C
C       REAL TIME CTD PLOTTING PROGRAM WHEN CTD USED IN TOWING MODE.
C       PLOTS S,T,P VS. TIME.  PLOT IS REINITIALIZED EVERY 30 MINUTES.
C       RELOCATABLES:  %LRSAL, %R, %POTMP
C
C       PROGRAMMER: LARRY ROSENBLUM
C       VERSION: SEPT 18, 1979
C
      DIMENSION DAT(3),NAMES(2),NAMET(2),NAMEP(2),NASTA(2),NALOW(2)
      DIMENSION NAMTM(4),NALBL(5),IPRAM(5),NAPOT(4)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      DATA NAMEP/2HPR,2HES/
      DATA NAMES/2HSA,2HL /
      DATA NAMET/2HTE,2HMP/
```

```
      DATA NAPOT/2HPO,2HT ,2HTE,2HMP/
      DATA NAMTM/2HTI,2HME,2H M,2HIN/
      DATA NASTA/2HST,2HA /
      DATA NALOW/2HLO,2HW /
      DATA NALBL/2HST,2HAR,2HT ,2HTI,2HME/
      CALL RMPAR(IPRAM)
      IANS1=IPRAM(1)
      CALL EXEC(22,0)
C       FIX AXES LENGTH AND SET TIME AXIS PARAMETERS
      SAXL=5.
      PAXL=5.
      TAXL=5.
      TMAXL=6.0
      TMIV=0.0
      TMINCR=5.0
 7    CALL PLTLU(7)
C       DRAW AXES, LABEL, AND ANNOTATE
      LR=1
      NSTIC=INT(SAXL)+1
      NPTIC=INT(PAXL)+1
      NTTIC=INT(TAXL)+1
      KNUM1=42
      KNUM2=45
      KNUM3=46
      LNUM1=5
      LNUM2=0
      LNUM3=3
      CALL AX(1.0,0.4,6.,7,4,0.0)
      CALL LABEL(1.0,0.2,.1,6.,7,TMIV,TMINCR,-1,0.0)
      CALL SYMB(3.4,0.0,.1,NAMTM,0.0,8)
      CALL AX(1.0,4.4,4.0,6,4,-90.0)
      CALL LABEL(0.9,4.4,.1,-4.,6,PIV,PINCR,2,90.)
      CALL SYMB(0.9,0.0,.1,KNUM1,90.,-1)
      CALL SYMB(0.1,0.5,.1,LNUM1,90.,-1)
      CALL SYMB(0.15,0.6,.1,NAMEP,90.,4)
      CALL LABEL(0.7,4.4,.1,-4.,6,SIV,SINCR,2,90.)
      CALL SYMB(0.7,0.0,.1,KNUM2,90.,-1)
      CALL SYMB(0.1,2.0,.1,LNUM2,90.,-1)
      CALL SYMB(0.15,2.1,.1,NAMES,90.,4)
      CALL LABEL(0.5,4.4,.1,-4.,6,TIV,TINCR,2,90.)
      CALL SYMB(0.5,0.0,.1,KNUM3,90.,-1)
      CALL SYMB(0.1,3.5,.1,LNUM3,90.,-1)
      IF(IANS1.EQ.1)CALL SYMB(0.15,3.6,.1,NAMET,90.,4)
      IF(IANS1.EQ.2)CALL SYMB(0.15,3.6,.1,NAPOT,90.,8)
      CALL SYMB(2.5,4.6,.1,NALBL,0.0,10)
      CALL SYMB(7.1,3.6,.1,NASTA,0.0,4)
      CALL SYMB(7.1,3.3,.1,NALOW,0.0,4)
      CALL AX(7.0,0.4,4.,6,4,90.)
      RIST=R(IST)
      RLOW=R(LOW)
      CALL LABEL(7.6,3.6,.1,0.,0.,RIST,0.0,-1,0.0)
      CALL LABEL(7.6,3.3,.1,0.,0.,RLOW,0.0,-1,0.0)
      CALL PLOT(1.0,0.4,-3)
      JREC=3
      KOUNT=0
C       SUSPEND
 30   CALL EXEC(6,0,1)
      IF(ISTOP.EQ.1.OR.ISTOP.EQ.3) GO TO 99
      IF(ISTOP.EQ.-2)GO TO 98
```

```
      IF(LR.NE.1) GO TO 32
      NDAT1=IBUF2(2)
      NDAT2=IBUF2(3)
      NTM1=IBUF2(2)/24
      NTM2=IBUF2(2)-NTM1*24
      NTM3=IBUF2(3)/60
      NTM4=IBUF2(3)-NTM3*60
      NDAT2=NDAT2-NTM4
      NTM4=0
      CALL NUMB(3.4,4.2,.1,FLOAT(NTM1),0.0,-1)
      CALL NUMB(3.9,4.2,.1,FLOAT(NTM2),0.0,-1)
      CALL NUMB(4.3,4.2,.1,FLOAT(NTM3),0.0,-1)
      CALL NUMB(4.7,4.2,.1,FLOAT(NTM4),0.0,-1)
32    NDEL1=IBUF2(2)-NDAT1
      NDEL2=IBUF2(3)-NDAT2
      IF(NDEL1.NE.0)NDEL2=NDEL2+3600
      IF(NDEL2.GT.1800) GO TO 98
      RE1=R(IBUF2(12))
      RE2=R(IBUF2(10))
      RE3=R(IBUF2(11))
      DAT(2)=C1*(RE2-C2)
      DAT(3)=C3+(C4*RE3)
      DAT(1)=C5*RE1*(1+(C6*DAT(2)))
C       CHECK TEMPERATURE SIGN
C     ISGN1=IAND(IBUF2(12),2B)
C        **** CTD HAS TROUBLE IN SIGN BYTE ****
C     IF(ISGN1.EQ.2)DAT(3)=-DAT(3)
C       DAT1=COND,DAT2=PRESS,DAT3=TEMP - ALL CONTVT TO ENG UNITS
C       CALL SALIN CONVERS SUBROUTINE
      CALL LRSAL(DAT(1),DAT(2),DAT(3),S)
      DAT(1)=S
C       IF IANS1=2 USE POTENTIAL TEMPERATURE
      IF(IANS1.EQ.2)CALL POTMP(DAT(2),DAT(3),DAT(1),PTMP)
      IF(IANS1.EQ.2)DAT(3)=PTMP
21    CALL PLT(DAT(1),DAT(3),DAT(2),NDEL2,LR)
      LR=LR+1
      GO TO 30
98    ISTOP=-1
      GO TO 7
99    END
      SUBROUTINE PLT(S,T,P,NDEL2,LR)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      WRITE(1,40)NDEL2,S,T,P
40    FORMAT(I8,3F10.4)
      SMAX=SIV+(SINCR*SAXL)
      TMAX=TIV+(TINCR*TAXL)
      PMAX=PIV+(PINCR*PAXL)
      XIV=60.*0.0
      XINCR=60.*5.
      XMAX=XIV+XINCR*6.
      IF(S.LT.SIV.OR.S.GT.SMAX)RETURN
      IF(T.LT.TIV.OR.T.GT.TMAX)RETURN
      IF(P.LT.PIV.OR.P.GT.PMAX)RETURN
      SSCAL=SAXL/(SMAX-SIV)
      SPLT=SSCAL*(SMAX-S)
```

```
      TSCAL=TAXL/(TMAX-TIV)
      TPLT=TSCAL*(TMAX-T)
      PSCAL=PAXL/(PMAX-PIV)
      PPLT=PSCAL*(PMAX-P)
      XDEL=R(NDEL2)
      XSCAL=6./(XMAX-XIV)
      XPLT=XSCAL*(XDEL-XIV)
      NUMB1=0
      NUMB2=3
      NUMB3=5
      SIZE=0.1
      IF(LR.NE.1)SIZE=0.03
C       SCALE Y AXIS FOR 2647 TERMINAL
      SPLT=0.8*SPLT
      TPLT=0.8*TPLT
      PPLT=0.8*PPLT
      CALL SYMB(XPLT,SPLT,SIZE,NUMB1,0.0,-1)
      CALL SYMB(XPLT,TPLT,SIZE,NUMB2,0.0,-1)
      CALL SYMB(XPLT,PPLT,SIZE,NUMB3,0.0,-1)
      RETURN
      END
      END$
```

```
FTN4,L
      PROGRAM CTDCN(3,14)
C
C       CONRTOL PROGRAM TO CAUSE END OR IDLE.
C       ISTOP CHANGED IN COMMON BY 'RU,CTDCN,,N   WHERE
C       N=-1 NORMAL CONDITION
C       N= 1 NORMAL END
C       N= 2 IDLE MODE
C       N= 3 ABORT
C       N=-2 BEGIN NEW PLOT
C       IBUF1&2(5) USED TO MARK SPECIAL FILE POINTS
C       USING IPRAM(1)
C
C        PROGRAMMER: LARRY ROSENBLUM
C        VERSION:  SEPT 22 1979
C
      DIMENSION IPRAM(5)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      CALL RMPAR(IPRAM)
      CALL EXEC(22,0)
      IF(IPRAM(2).EQ.0)IPRAM(2)=-1
      ISTOP=IPRAM(2)
      IBUF1(5)=IPRAM(1)
      IBUF2(5)=IPRAM(1)
      END
      END$
```

```
FTN4,L
      PROGRAM CTDMS
      DIMENSION IPRAM(5),IBACK(5)
      CALL RMPAR(IPRAM)
      IF(IPRAM(1).EQ.2)GO TO 20
      IF(IPRAM(1).EQ.3)GO TO 30
5     WRITE(1,10)
10    FORMAT("PLOT TYPE? 1=S,T VS. P (LOWERING)",/
     1       "              2=S,T,P VS TIME (TOWING)")
      READ(1,*)IANS
      IF(IANS.NE.1.AND.IANS.NE.2) GO TO 5
12    WRITE(1,15)
15    FORMAT("TEMPATURE TYPE: 1=TEMP., 2=POTENTIAL TEMP.")
      READ(1,*)IANS1
      IF(IANS1.NE.1.AND.IANS1.NE.2)GO TO 12
      IBACK(2)=IANS
      IBACK(3)=IANS1
      CALL PRTN(IBACK)
      CALL EXEC(6)
20    WRITE(1,25)
25    FORMAT("PAUSE - TYPE: 'GO,CTDRT' FOR DATA AQUISITION")
      GO TO 99
30    WRITE(1,35)
35    FORMAT("PAUSE - IDLE MODE - TYPE: 'GO,CTDRT' TO CONTINUE")
99    END
      END$
```

```
FTN4,L
      PROGRAM CTDFL(3,16)
C
C  THIS PROGRAM SETS THE FLOROMETER ON/OFF SWICTH.
C  IFLOR=0 FLOROMETER OFF
C  IFLOR=1 FLOROMETER ON
C  TYPE: 'RU,CTDFL,N' WHERE N=2 FOR OFF OR N=3 FOR ON
C
C  PROGRAMMER: LARRY ROSENLUM
C  VERSION: SEPT. 20, 1979
      DIMENSION IPRAM(5)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      CALL RMPAR(IPRAM)
      IF(IPRAM(1).EQ.2)IFLOR=0
      IF(IPRAM(1).EQ.3)IFLOR=1
      IF(IPRAM(1).NE.2.AND.IPRAM(1).NE.3)WRITE(1,10)
10    FORMAT("TYPE 'RU,CTDFL,N' WHERE N=2 (OFF) OR =3(ON)")
      END
      END$
```

```
FTN4,L
      PROGRAM CTDTR(3,90)
C
C       PROGRAM TO WRITE TERMINATION FILE AND OPEN PLOTTING PROG. SO
C       THAT IT GETS CLOSED CORRECTLY.  TERM IN SCHEDULED BY CTDRT
C       WHEN ISTOP= 1 OR 3 - SEE CTDCN.
C
C       FORMAT OF 8 WORD TERMINATION FILE:
C       WORD 1       -1
C       WORDS 2&3  TIME IN HOURS AND SECONDS
C       WORD 4       ISTOP/3
C       WORDS 5-8    0
C
C       PROGRAMMER: LARRY ROSENBLUM
C       VERSION:  SEPT. 20, 1979
C
      DIMENSION IBUF4(8),ITIME(5),NAME1(3),NAME2(3),IPRAM(5)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
      DATA NAME1/2HCT,2HDP,2H1 /,NAME2/2HCT,2HDP,2H2 /
C       GET PLOT VERSION PARAMETER
      CALL RMPAR(IPRAM)
      IANS=IPRAM(1)
      CALL EXEC(22,0)
C       WRITE EOF AFTER DATA
      CALL EXEC(3,8+100B)
C       SCHEDULE PLOTTING PROGRAM
      IF(IANS.EQ.1)CALL EXEC(10,NAME1)
      IF(IANS.EQ.2)CALL EXEC(10,NAME2)
C       WRITE TERMINATION FILE AND DBL EOF
      IBUF4(1)=-1
      CALL EXEC(11,ITIME)
      IBUF4(2)=24*ITIME(5)+ITIME(4)
      IBUF4(3)=60*ITIME(3)+ITIME(2)
      IBUF4(4)=ISTOP/3
      IBUF4(5)=0
      IBUF4(6)=0
      IBUF4(7)=0
      IBUF4(8)=0
      CALL EXEC(2,8,IBUF4,8)
      CALL EXEC(3,8+100B)
      CALL EXEC(3,8+100B)
      END
      END$
```

```
      FTN,L
            PROGRAM F2EOF
      C
      C           VERSION 7/3/77
      C           DEAN CLAMONS
      C
      C           THIS PROGRAM FINDS THE FIRST DOUBLE EOF ON A TAPE
      C           AND POSITIONS THE TAPE AFTER THE FIRST OF THE PAIR
      C           THE METHOD IS TO BACK UP TWO RECORDS (IN CASE WE
      C           ARE ALREADY AT THE EOF'S) AND THEN SPACE FORWARD
      C           A FILE AND READ THE NEXT RECORD.
      C           IF IT IS AN EOF WE BACK UP ONE RECOR AND ARE DONE
      C           IF NOT AN EOF WE SPACE FORWARD ANOTHER FILE
      C           AND REPEAT
      C
      C           NOTE!!!!!!
      C             THE TAPE MUST HAVE A DOUBLE EOF FOR THIS
      C             PROGRAM TO WORK. IF NOT IT WILL RUN AWAY
      C
            LMT=8
            CALL EXEC(3,200B+LMT)
            CALL EXEC(3,200B+LMT)
          1 CALL EXEC(3,1300B+LMT)
            CALL EXEC(1,LMT,IBFF,1)
            IF(IEOF(LMT).GE.0)GO TO 1
            CALL EXEC(3,200B+LMT)
            END
            END$
```

USER'S GUIDE TO THE REAL-TIME CTD PROGRAMS

Preparation

Prior to starting the CTD data collection process the operator should ensure that the system is set for CTD data. The operator should:

1. Check that power is on to the CTD deck unit and the unit is displaying reasonable values.

2. Toggle the "hold" switch on the CTD deck unit several times to assure that the sync frame is good. The sync frame indicator should display either 15 or 240 (decimal).

3. Check that the Computer Applications Group CTD Interface has been switched to the "ON" position.

4. If the computer system has been performing other functions, assure that the logical units (LU) assignments are correct. The CTD programs expect that LU-1 is the system console, LU-6 is the line printer, LU-7 is the terminal plotter, LU-8 is the magnetic tape unit, and LU-10 is the CTD. If necessary, logical unit assignments should be changed using the RTE operator "LU" command, so that these logical units will point to the appropriate equipment table entry.

5. Mount a tape on the tape drive and check that the subchannel indicator on the tape unit is set to 0 and that the tape is "write" enabled (the write ring is installed).

6. Obtain the latitude and longitude from a laboratory display, a computer program for this purpose, or the navigation laboratory.

If a CTD lowering (or tow) has just been completed, then, to begin the next lowering (or tow), the operator should get an RTE system prompt * and type "RU, F2EOF." This program will position the tape between the two EOF's which follow the termination record of the previous CTD. Perform step 6 above.

Program Start-Up

The next two sections of this chapter describe the interaction between the operator and CTD real-time programs. For purposes of identification within this report only, all user responses are underlined. All user entries are to be followed by a carriage return. The symbol * is the RTE system prompt and may be obtained in an RTE system by striking any key on the console one time. Answers to questions are by way of example.

```
* RU, CTDRT
ENTER CTD MODE, 1 = LOWERING, 2 = TOWING
1
ENTER TEMPERATURE TYPE, 1 = TEMP, 2 = POTENTIAL TEMP
2
ENTER SHIP'S NAME (CRUISE) (UP TO 12 CHARACTERS)
HAYES 9999
ENTER DATE (YR, MO, DAY)
1979 10 8
ENTER STATION NUMBER & LOWERING NUMBER
2 1
ENTER LATITUDE: DEG, MIN, & FRACTION OF MIN xxxx., xx.xxx
```

32. 15.25
ENTER LONGITUDE: DEG, MIN. & FRACTION OF MIN xxxx., xx.xxx
-65. 22.23

At this point the operator is asked
USE DEFAULT VALUES, 1 = YES, 2 = NO

The default values concern the instrument number, the values of the constants $C1$, $C2$, ..., $C6$ used to convert readings into pressure, temperature, and conductivity and the axes limits for the real-time plots. The conversion constant default values are: $C1 = 0.1$, $C2 = 0.0$, $C3 = 0.0$, $C4 = 0.0005$, $C5 = 0.001$, and $C6 = 0.0$. The only constant with which the operator need be concerned is $C1$, which is dependent on the depth rating of the pressure sensor. The default is for the 6,500-decibar (65-MPa) pressure sensor. If a different pressure sensor is used, $C1$ should be changed to the value shown in Table 1.

Table 1—Values of the Conversion Constant $C1$ for Each of the Different CTD Pressure Sensors

Pressure-Sensor Range (decibar)	(MPa)	$C1$ Value
0–6500	0–65	0.1 (default)
0–3200	0–32	0.05
0–1600	0–16	0.025
0–650	0–6.5	0.01

The axes lengths are fixed at 6 inches (15.2 cm) for the x axis and 5 inches (12.7 cm) for the y axis. The operator supplies the initial value and the increments per inch for each parameter—pressure, temperature, and salinity. The default values for lowering plots are given in Table 2 and for towing plots in Table 3. In general, except for deep-water lowerings, the operator will want to specify more limited axes scaling using knowledge of the area and depth involved. Returning to the question about default values, let us assume we wish to lower a 1600-decibar (16-MPa) pressure sensor from 200 to 500 decibars (2 to 5 MPa) and that we know that temperature will be in the range 14 to 19°C and salinity in the range 34.5 to 36.0 mmho(mS). In the following the symbol "□ CR" will denote a single space followed by a carriage return. Then, continuing our sample question and answer, the programs would ask:

USE DEFAULT VALUES, 1 = YES, 2 = NO
2
DEFAULT VALUES GIVEN IN PARENTHESIS
INSTRUMENT # (1)
□ CR
CAL CONST PRESSURE C1, C2 (0.1, 0.0)
0.025 0.0
CAL CONST TEMP C3, C4 (0.0, 0.0005)
□ CR
CAL CONST C5, C6 (0.001, 0.0)
□ CR
FOR EACH AXIS THE AXIS LENGTH IS GIVEN IN PARENTHESIS ENTER
THE INITIAL VALUE AND THE INCREMENT/INCH.
PRESSURE (AX LNTH = 5) ENTER INIT VALUE, INCR INCH
200.0 60.0
TEMP (AX LNTH = 6) ENTER INIT VALUE, INCR/INCH
14.0 1.0
SAL (AX LNTH = 6) ENTER INIT VALUE, INCR/INCH
34.5 0.25

Table 2—Plot Default Values for the Lowering Mode

Parameter	Axis Length (in.)	(cm)	Initial Value	Incr (per in.)	(per cm)	Range
Pressure	5.0	12.7	0	1000	394	0–5000 dbar (0–50 MPa)
Temperature	6.0	15.2	1	4	1.6	1–25 °C
Salinity	6.0	15.2	32	1	0.4	32–38 mS

Table 3 - Plot Default Values for the Towing Mode

Parameter	Axis Length (in.)	cm	Initial Value	Incr (per in.)	(per cm)	Range
Pressure	5.0	12.7	0	10	3.9	0 – 50 dbar
Temperature	5.0	12.7	1	4	1.6	(0 – 0.5 MPa)
Salinity	5.0	12.7	33	1	0.4	33 – 38 mS
Time	6.0	15.2	Start time of plot	5	2.0	0 – 30 min

The operator entries are printed out on the terminal and written to the line printer. The operator is then given the option of restarting the question process if any mistakes have been made (the answer "3" to the following question is for debug purposes only and should not be used).

ARE INPUT PARAMS O.K.: 1 = YES, 2 = NO, 3 = YES – NO MT FILE
<u>1</u>
PAUSE—TYPE 'GO, CTDRT' FOR DATA ACQUISITION

At this point the plot axes and annotation will be output on the plotting terminal. The operator should verify that these are correct, especially the parameter limits on each axis. Once the plot axes and annotation are drawn the program is prepared to acquire data. The operator now obtains the system prompt (*) and types:

* <u>GO, CTDRT</u>

This completes the start-up procedure.

Special Functions

The program CTDCN is used to

- end data taking,
- abort data taking,
- put the program into an "idle" mode, or
- mark special segments.

The end and abort produce exactly the same result, except that a word in the termination file indicates which way the program was terminated. The idle mode is used to suspend data collection temporarily and the special segment marker changes one word in the data buffers. This last operation can be used to identify data segments for later processing. These operations occur when the operator types:

for a NORMAL TERMINATION
* <u>RU, CTDCN,,1</u>
for an ABORT
* <u>RU, CTDCN,,3</u>
for an IDLE
* <u>RU, CTDCN,,2</u>

The message PAUSE—IDLE MODE—TYPE: 'GO, CTDRT' TO CONTINUE will appear on the system console. To continue the collection process the operator types:

* GO, CTDRT

for marking SPECIAL SEGMENTS

* RU, CTDCN,N

where *N* is any number. This number is placed in Word 5 of each data buffer until it is changed or the programs are terminated.

The program CTDFL sets the fluorometer on/off switch.

The command

* RU,CTDFL, 3

turns the switch on (fluorometer data being taken) and the command

* RU, CTDFL, 2

turns the switch off. It should be noted that the switch is initialized as "off" and that it is used only as a marker; failure to set the switch does not alter the data.

DATA-PROCESSING PROGRAMS

Parameters of Interest

Once a CTD data tape has been collected, many parameters of interest to the oceanographic researcher can be computed from the stored values of conductivity, temperature, and pressure. To assist the researcher a program called PLTBK has been written which will compute values and produce plots of various parameters. To do this the user is offered the following parameter list:

1 = Salinity
2 = Pressure
3 = Temperature
4 = Potential Temperature
5 = Sound Velocity
6 = Density
7 = Specific Volume
8 = Adiabatic Temperature Gradient
9 = Sigma (0)
10 = Depth
11 = Brunt-Vaisala Frequency
12 = Time (minutes)
13 = d(temp) / d(pressure)
14 = d(salinity) / d(pressure)
15 = Conductivity
16 = Fluorometer

Program PLTBK allows the user to compute and plot (against each other) any pair of the above parameters or, alternately, to define any one parameter as the y axis and plot any two variables as x-axis parameters against the selected y-axis parameters. Thus, for example, if the user wanted to obtain a sound-speed profile the user would select 5(sound velocity) for the x axis and 10(depth) for the y axis. To obtain a plot of temperature and conductivity vs pressure, the user would select 3(temperature) for

the x axis, 2(pressure) for the y axis and 15(conductivity) as a second x axis. The axes scales for the plot are user selected. Scientific subroutines for calculating these parameters are listed in the appendix. These subroutines were provided by the Woods Hole Oceanographic Institution, except for the Brunt-Vaisala frequency calculation, which uses an approximation formula and is valid for the top several hundred meters.

User's Guide to Program PLTBK

This section describes the interaction between the user and program PLTBK. As before, user responses will be underlined and answers are by way of example.

* RU, PLTBK
ENTER STATION NUMBER AND LOWERING NUMBER
2 1
DO YOU WANT A TIME SEARCH? 1=YES, 2=NO

A time search allows the user to look at only part of an event; otherwise the entire event will be processed.

1
ENTER START TIME—DAY HR MIN SEC
171 5 27 0
ENTER STOP TIME—DAY HR MIN SEC
171 6 31 0

1=SALINITY	2=PRESSURE	3=TEMPERATURE
4=POT. TEMP	5=SOUNDVEL	6=DENSITY
7=SPEC VOL	8=ATG	9=SIGMA
10=DEPTH	11=B-V FREQ	12=TIME
13=D(T) /D(P)	14=D(S) /D(P)	15=CONDUCTIVITY
16=FLUOROMETER		

ENTER X AXIS PARAMETER NUMBER
3
ENTER Y AXIS PARAMETER NUMBER
10
DO YOU WANT A SECOND X AXIS, 1=YES, 2=NO
1
ENTER X2 AXIS PARAMETER NUMBER
15
ENTER X AXIS LENGTH, INIT VALUE, INCR/INCH
6 3. 15.
ENTER Y AXIS LENGTH, INIT VALUE, INCR/INCH
5 2000. 400.
ENTER X2 AXIS LENGTH, INIT VALUE, INCR/INCH
6 32. .5

Since using all points usually provides too dense a plot, the user chooses the plotting density by answering the following question:

OF POINTS/4 SEC SCAN; MUST DIVIDE 64
8
The program will now plot the data and stop.

Examples of Program PLTBK Graphics

In this section, several examples of program PLTBK plots are given. The plots were produced using PLTBK from a digital tape recorded on the USNS *Lynch* in April 1978 in the Atlantic off the continental shelf of the United States. Figure 6 shows salinity and temperature plotted against pressure for

the full lowering (approx. 100 m to 4600 m) plotting one point every 4 seconds. Figure 7 shows salinity for the top 1000 meters, plotting one point every second. Figure 8 is the sound-velocity profile for the time and site. Figure 9 is a temperature-salinity plot for the lowering. Figure 10 is a density plot and Figure 11 plots density vs potential temperature. Illustrating the ability to plot any two parameters against any other, Figure 12 shows salinity and temperature plotted against sound speed.

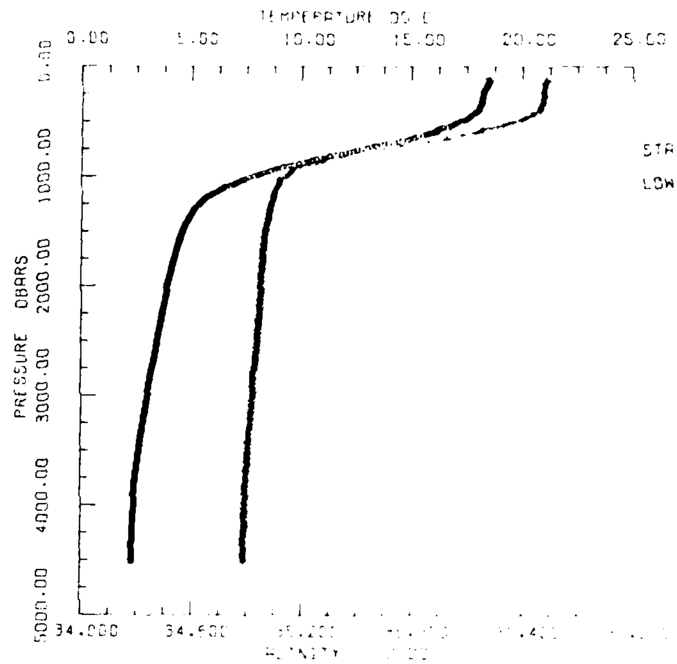


Figure 6 — Plot of salinity and temperature vs pressure for the entire lowering

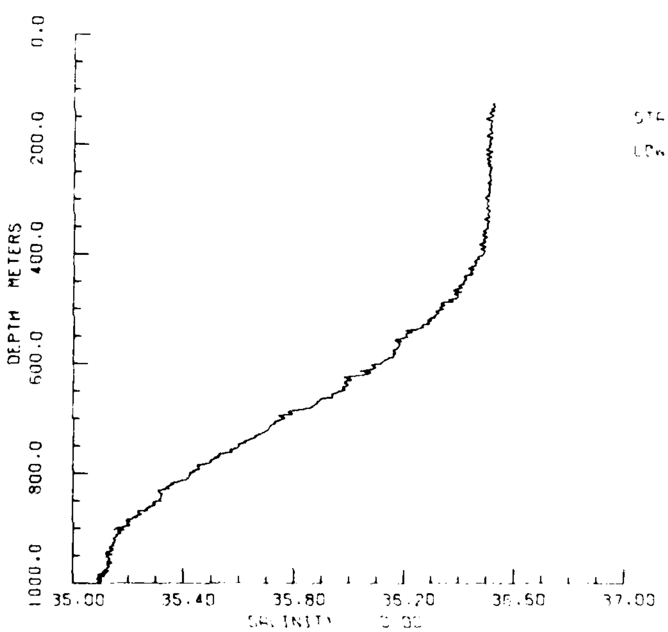


Figure 7 — Expanded salinity plot for top 1000 m

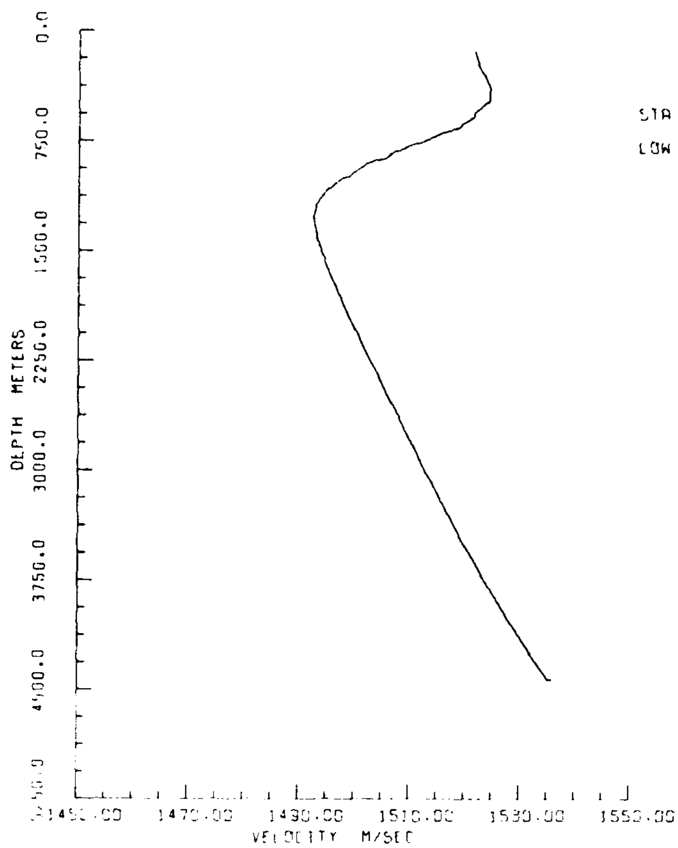


Figure 8 — Sound velocity profile plot for site

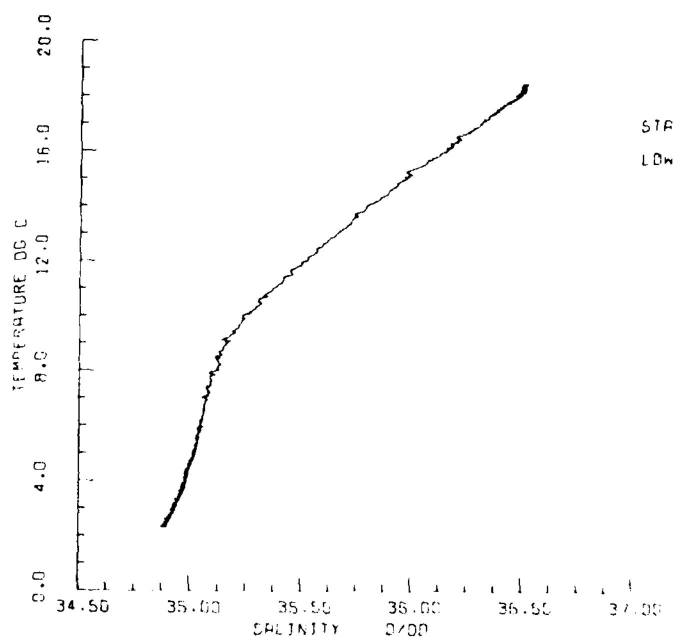


Figure 9 — Temperature - salinity plot for the lowering

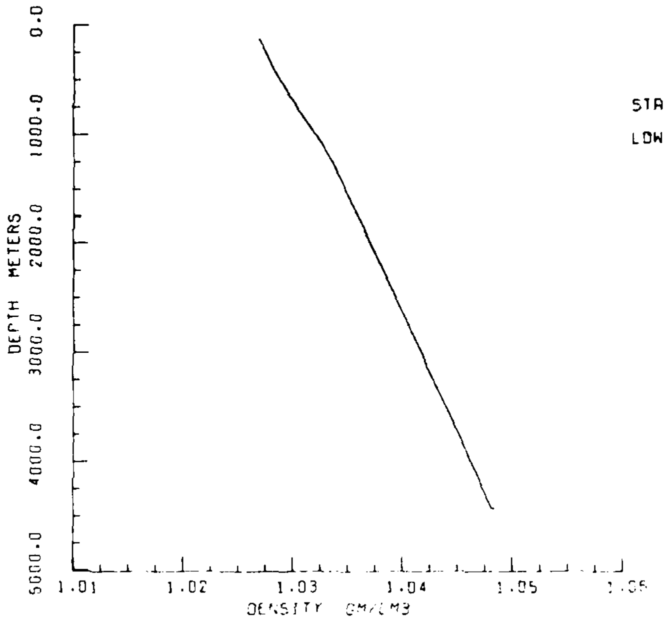


Figure 10 — Density plot for the lowering

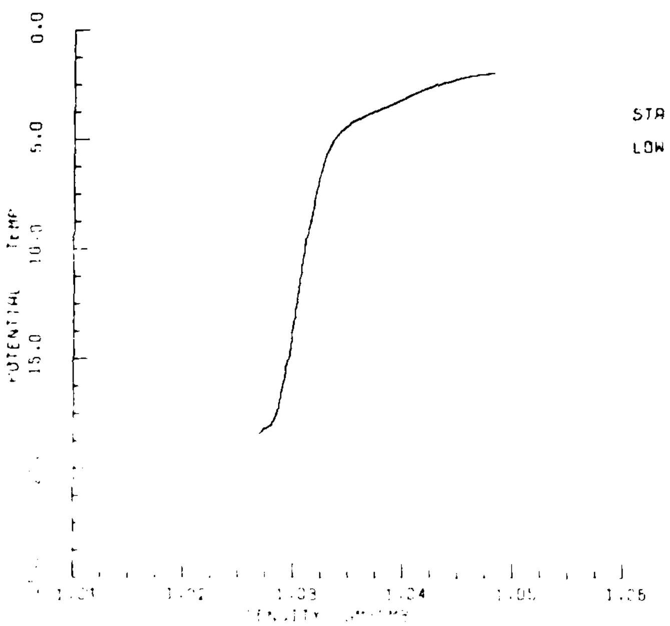


Figure 11 — Plot of density vs potential temperature

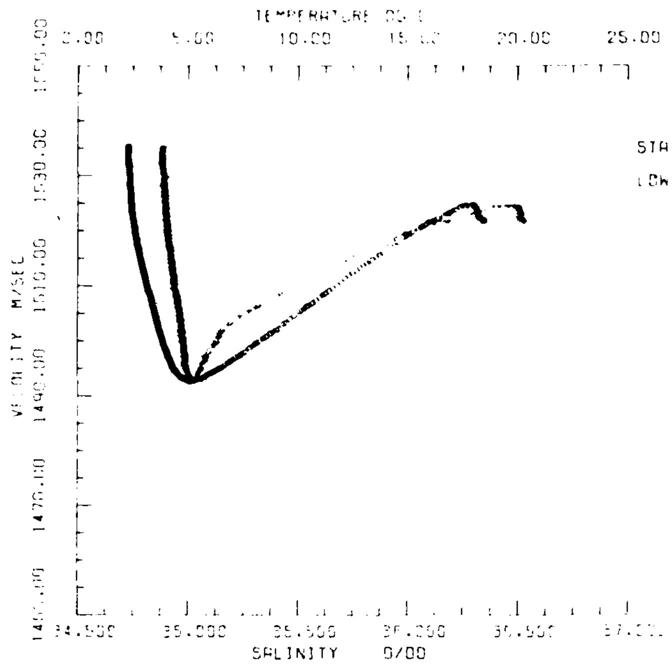


Figure 12 — Plot of salinity and temperature vs sound velocity

Data Processing Programs—Fortran Source Listings

```
FTN4,L
      PROGRAM PLTBK
C
C       THIS VERSION REVISED TO WORK FOR FLUOROMETER DATA SET 10 /79
C       THIS PROGRAM WILL PRODUCE PLOTS FROM THE COLLECTED CTD DATA
C       TAPE.  ANY PARAMETER FROM THE PARAMETER LIST CAN BE PLOTTED
C       AGAINST ANY OTHER PARAMETER OR AGAINST ANY PAIR OF PARAMETERS
C       FROM THE LIST.  TAPE MUST BE POSITIONED AT THE DATA.  SCIENTIFIC
C       SUBROUTINES USED WERE SUPPLIED BY W.H.O.I. EXCEPT FOR THE APPROX.
C       USED FOR BRUNT-VAISALA FREQUENCY.
C
C       RELOCATABLE SUBROUTINES: %LRSAL,%R,%POTMP,%SVEL,%ATG,%SIGMT,
C                                %SPVOL,%BRVA1
C
C       THIS VERSION ADDS THE FLUOROMETER.
C
C       PROGRAMMER: LARRY ROSENBLUM     VERSION: SEPT. 1979
C
      DIMENSION DAT(16),NAMEP(8),NAMES(8),NAMET(8),NAMPT(8),NAME(8,16)
      DIMENSION NAMEV(8),NASTA(2),NALOW(2),NADEN(8),NASPV(8),NAATG(8)
      DIMENSION NASMT(8),NADPH(8),NABVF(8),NATME(8),NALBL(5),NADLT(8)
      DIMENSION NADLS(8),NAFLR(8),NACON(8)
      DIMENSION IREG(2),XBV(500),IAX(3)
      DOUBLE PRECISION XALPH,XBV,DENS,SUMRH,XSO,XST
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
```

```
      COMMON X1AXL,X1IV,X1INCR,YAXL,YIV,YINCR,X2AXL,X2IV,X2INCR
      EQUIVALENCE (NAMES,NAME(1,1)),(NAMEP,NAME(1,2))
      EQUIVALENCE (NAMET,NAME(1,3)),(NAMPT,NAME(1,4))
      EQUIVALENCE (NAMEV,NAME(1,5)),(NADEN,NAME(1,6))
      EQUIVALENCE (NASPV,NAME(1,7)),(NAATG,NAME(1,8))
      EQUIVALENCE (NASMT,NAME(1,9)),(NADPH,NAME(1,10))
      EQUIVALENCE (NABVF,NAME(1,11)),(NATME,NAME(1,12))
      EQUIVALENCE (NADLT,NAME(1,13)),(NADLS,NAME(1,14))
      EQUIVALENCE (NACON,NAME(1,15)),(NAFLR,NAME(1,16))
      EQUIVALENCE (REG,IREG)
      DATA NAMEP/2HPR,2HES,2HSU,2HRE,2H  ,2HDB,2HAR,2HS /
      DATA NAMES/2HSA,2HLI,2HNI,2HTY,2H  ,2H  ,2H0/,2H00/
      DATA NAMET/2HTE,2HMP,2HER,2HAT,2HUR,2HE ,2HDG,2H C/
      DATA NAMPT/2HPO,2HTE,2HNT,2HIA,2HL ,2H  ,2HTE,2HMP/
      DATA NAMEV/2HVE,2HLO,2HCI,2HTY,2H  ,2HM/,2HSE,2HC /
      DATA NADEN/2HDE,2HNS,2HIT,2HY ,2H G,2HM/,2HCM,2H3  /
      DATA NASPV/2HSP,2H. ,2HVO,2HL.,2H  ,2HCM,2H3/,2HGM/
      DATA NAATG/2HAD,2HIA,2HBT,2H T,2HEM,2HP ,2HGR,2HAD/
      DATA NASMT/2HSI,2HGM,2HA ,2H T,2H  ,2HGM,2H/C,2HM3/
      DATA NADPH/2H  ,2HDE,2HPT,2HH ,2H M,2HET,2HER,2HS /
      DATA NABVF/2HBR,2H-V,2HA ,2HFR,2HEQ,2H C,2HY/,2HHR/
      DATA NATME/2H  ,2HTI,2HME,2H  ,2H  ,2HMI,2HN ,2H  /
      DATA NADLT/2H  ,2HDE,2HL(,2HT),2H/D,2HEL,2H(P,2H) /
      DATA NADLS/2H  ,2HDE,2HL(,2HS),2H/D,2HEL,2H(P,2H) /
      DATA NACON/2H  ,2HCO,2HND,2HUC,2HTI,2HVI,2HTY,2H  /
      DATA NAFLR/2H  ,2H F,2HLO,2HRO,2HME,2HTE,2HR ,2H  /
      DATA NASTA/2HST,2HA /
      DATA NALOW/2HLO,2HW /
      DATA NALBL/2HST,2HAR,2HT ,2HTI,2HME/
C
      LUMT=8
      POLD=0.0
      WRITE(1,111)
  111 FORMAT("ENTER STATION NUMBER AND LOWERING NUMBER")
      READ(1,*)IST,LOW
      WRITE(1,200)
  200 FORMAT("DO YOU WANT A TIME SEARCH? 1=YES, 2=NO")
      READ(1,*)IANS1
      IF(IANS1.NE.1) GO TO 206
      WRITE(1,202)
  202 FORMAT("ENTER START TIME - DAY HRS MIN SECS")
      READ(1,*)IST1,IST2,IST3,IST4
      WRITE(1,204)
  204 FORMAT("ENTER STOP TIME - DAY HRS MIN SECS")
      READ(1,*)ISP1,ISP2,ISP3,ISP4
      ISP5=24*ISP1+ISP2
      IST5=24*IST1+IST2
      IST6=60*IST3+IST4
      ISP6=60*ISP3+ISP4
  206 WRITE(1,12)
   12 FORMAT(" 1=SALINITY     2=PRESSURE     3=TEMPERATURE",/
     1       " 4=POT. TEMP    5 =SOUND VEL.  6=DENSITY",/
     2       " 7=SPEC. VOL.   8=ATG          9=SIGMA T(MOD)",/
     3       "10=DEPTH       11=B-V FREQ.   12=TIME(MIN)",/
     4       "13=D(T)/D(P)   14=D(S)/D(P)   15=CONDUCTIVITY",/
     5       "16=FLOROMETER")
      WRITE(1,101)
  101 FORMAT("ENTER X AXIS PARAMETER NUMBER")
      READ(1,*)IAX(1)
```

```
      WRITE(1,102)
102   FORMAT("ENTER Y AXIS PARAMETER NUMBER")
      READ(1,*)IAX(2)
      WRITE(1,105)
105   FORMAT("DO YOU WANT A 2ND X AXIS: 1=YES, 2=NO")
      READ(1,*)IOPT
      IF(IOPT.EQ.2) GO TO 123
      WRITE(1,106)
106   FORMAT("ENTER X2 AXIS PARAMETER NUMBER")
      READ(1,*)IAX(3)
123   IF(IAX(1).NE.11.AND.IAX(3).NE.11.AND.IAX(2).NE.11)GO TO 113
      WRITE(1,127)
127   FORMAT("BE SURE THAT B-V FREQ. IS PLOTTED VS. DEPTH ONLY")
      WRITE(1,124)
124   FORMAT("ENTER B-V FREQ.DEPTH BIN SIZE, BETWEEN 1M. & 20 M.")
      READ(1,*)AVDPT
      IF(AVDPT.LT.1.0.OR.AVDPT.GT.20.0)WRITE(1,126)
126   FORMAT("FREQ. BIN NOT WITHIN REQUIRED RANGE")
      IF(AVDPT.LT.1.0.OR.AVDPT.GT.20.0)GO TO 123
      WRITE(1,125)
125   FORMAT("ENTER SURFACE DENSITY-OBTAIN FROM DENSITY PLOT")
      READ(1,*)RHO0
C       INITIALIZE BRUNT-VAISALA AVERAGING COUNTERS
       NUMBV=1
      DPTBV=0.0+AVDPT
      DLAST=0.0
113   IF(IAX(1).NE.12.AND.IAX(3).NE.12.AND.IAX(2).NE.12.) GO TO 107
      WRITE(1,110)
110    FORMAT("ENTER PLOT START TIME: DAYS,HRS,MIN,SECS")
      READ(1,*)NT1,NT2,NT3,NT4
      NDAT1=24*NT1+NT2
      NDAT2=60*NT3+NT4
 107  WRITE(1,103)
103   FORMAT("ENTER:X AXIS LENGTH, INIT. VALUE, INCR/INCH")
      READ(1,*)X1AXL,X1IV,X1INCR
      WRITE(1,104)
104   FORMAT("ENTER: Y AXIS LENGTH, INIT. VALUE, INCR/INCH")
      READ(1,*)YAXL,YIV,YINCR
      IF(IOPT.EQ.2) GO TO 109
      WRITE(1,108)
108   FORMAT("ENTER: X2 AXIS LENGTH, INIT. VALUE, INCR/INCH")
      READ(1,*)X2AXL,X2IV,X2INCR
109   CALL INIT1
      WRITE(1,11)
11    FORMAT("# OF PTS/4 SEC SCAN;  MUST DIVIDE  2**7")
      READ(1,*)INUM
      JSKIP=448/INUM
      CALL PLTLU(7)
      NYTIC=INT(YAXL)+1
      NX1TIC=INT(X1AXL)+1
      NX2TIC=INT(X2AXL)+1
      CALL AX(0.6,0.4,X1AXL,NX1TIC,3,0.0)
      CALL AX(0.6,YAXL+0.4,YAXL,NYTIC,3,-90.)
      CALL LABEL(0.6,0.2,.1,X1AXL,NX1TIC,X1IV,X1INCR,3,0.0)
      CALL SYMB(-.3+X1AXL/2.,0.0,.1,NAME(1,IAX(1)),0.0,16)
      CALL LABEL(0.3,YAXL+0.4,.1,-YAXL,NYTIC.YIV,YINCR,2,+90.)
      CALL SYMB(0.1,-.3+YAXL/2.,.1,NAME(1,IAX(2)),90.0,16)
      CALL SYMB(.7+X1AXL,-.4+YAXL,.1,NASTA,0.0,4)
```

```
      CALL SYMB(.7+X1AXL,-.7+YAXL,.1,NALOW,0.0,4)
      RIST=R(IST)
      RLOW=R(LOW)
      CALL LABEL(1.2+X1AXL,-.4+YAXL,.1,0.,0.,RIST,0.0,-1,0.0)
      CALL LABEL(1.2+X1AXL,-.7+YAXL,.1,0.,0.,RLOW,0.0,-1,0.0)
      IF(IOPT.EQ.2) GO TO 19
      CALL AX(.6+X2AXL,.4+YAXL,X2AXL,NX2TIC,3,-180.)
      CALL LABEL(.6,.6+YAXL,.1,X2AXL,NX2TIC,X2IV,X2INCR,2,0.)
      CALL SYMB(-.3+X2AXL/2.,.8+YAXL,.1,NAME(1,IAX(3)),0.0,16)
   19 CALL PLOT(0.6,0.4,-3)
      IF(IAX(1).NE.12.AND.IAX(2).NE.12.AND.IAX(3).NE.12) GO TO 15
      CALL SYMB(2.1,5.2,.1,NALBL,0.0,10)
      CALL NUMB(3.4,5.2,.1,FLOAT(NT1),0.0,-1)
      CALL NUMB(3.9,5.2,.1,FLOAT(NT2),0.0,-1)
      CALL NUMB(4.3,5.2,.1,FLOAT(NT3),0.0,-1)
      CALL NUMB(4.7,5.2,.1,FLOAT(NT4),0.0,-1)
C       READ IN DATA
15    REG= EXEC(1,LUMT,IBUF2,456)
      ILEN=IREG(2)
      IF(IANS1.NE.1) GO TO 208
      IF(IEOF(LUMT).LT.0) GO TO 15
      IF(IBUF2(2).LT.IST5) GO TO 15
      IF(ILEN.NE.456) GO TO 15
      IF(IBUF2(2).EQ.IST5.AND.IBUF2(3).LT.IST6) GO TO 15
      IF(IBUF2(2).GT.ISP5)GO TO 99
      IF(IBUF2(2).EQ.ISP5.AND.IBUF2(3).GT.ISP6) GO TO 99
208   IF(IEOF(LUMT).LT.0) GO TO 99
      DO 20 I=9,456,JSKIP
      RE1=R(IBUF2(I+3))
      RE2=R(IBUF2(I+1))
      RE3=R(IBUF2(I+2))
C
C       CALCULATE CONDUCTIVITY, PRESSURE, AND TEMPERATURE
C       DAT1=COND, DAT2=PRESS, DAT3=TEMP
C       THEN CONVERT TO SALINITY AND PUT DAT1=SAL
C       STORE COND IN DAT(15)
C
      DAT(2)=C1*(RE2-C2)
      DAT(3)=C3+(C4*RE3)
      DAT(1)=C5*RE1*(1+(C6*DAT(2)))
C     GET SIGN OF TEMPERATURE
C         **** MAY BE PROBLEM IN CTD WITH STATUS BIT ****
C     ITST1=IAND(IBUF2(I+4),2B)
C     IF(ITST1.EQ.2) DAT(3)=-DAT(3)
      DAT(15)=DAT(1)
      CALL LRSAL(DAT(1),DAT(2),DAT(3),S)
      DAT(1)=S
      WRITE(6,992)DAT(2),DAT(3),DAT(1)
992   FORMAT(1X,3(F12.2,2X))
C
C       CALCULATE FLOROMETER READING
      IF(IAX(1).NE.16.AND.IAX(2).NE.16.AND.IAX(3).NE.16)GO TO 21
      DAT(16)=FLOAT(IBUF2(I+5))*FLOAT(IBUF2(I+6))
C
C       CALCULATE POTENTIAL TEMPERATURE
C
21    IF(IAX(1).NE.4.AND.IAX(2).NE.4.AND.IAX(3).NE.4) GO TO 24
      CALL POTMP(DAT(2),DAT(3),DAT(1),PTMP)
      DAT(4)=PTMP
```

```
C
C       CALCULATE SOUND VELOCITY
C
24    IF(IAX(1).NE.5.AND.IAX(2).NE.5.AND.IAX(3).NE.5) GO TO 26
      CALL SVEL(DAT(2),DAT(3),DAT(1),DAT(5))
C
C       CALCULATE SPECIFIC VOLUME AND DENSITY
C
26    IF(IAX(1).EQ.7.OR.IAX(3).EQ.7.OR.IAX(2).EQ.7) GO TO 29
      IF(IAX(1).EQ.6.OR.IAX(3).EQ.6.OR.IAX(2).EQ.6) GO TO 29
      IF(IAX(1).EQ.9.OR.IAX(3).EQ.9.OR.IAX(2).EQ.9) GO TO 29
      IF(IAX(1).EQ.11.OR.IAX(3).EQ.11.OR.IAX(2).EQ.11) GO TO 29
      GO TO 28
29    CALL SIGMT(DAT(3),DAT(1),XS0,XST)
      CALL SPVOL(DAT(2),DAT(3),XS0,XST,XALPH)
      DAT(7)=XALPH
      DENS=1.0/XALPH
      DAT(6)=1./DAT(7)
      DAT(9)=1000.*(DAT(6)-1.0)
C
C       CALCULATE ADIABATIC TEMPERATURE GRADIENT
C
28    IF(IAX(1).NE.8.AND.IAX(3).NE.8.AND.IAX(2).NE.9) GO TO 30
      CALL ATG(DAT(2),DAT(3),DAT(1),GAMMA)
      DAT(8)=GAMMA
C
C       CALCULATE DEPTH FROM PRESSURE
C
30    IF(IAX(1).EQ.10.OR.IAX(2).EQ.10.OR.IAX(3).EQ.10) GO TO 32
      IF(IAX(1).EQ.11.OR.IAX(2).EQ.11.OR.IAX(3).EQ.11)GO TO 32
      GO TO 31
32    IF(DAT(2).LE.600.0)DAT(10)=DAT(2)/1.009
      IF(DAT(2).LE.600.0)GO TO 31
      A=.980665*2.022365
      A2=1.E-6
      A=A*A2
      B=.980665*1.02940
      C=-DAT(2)
      ROOTZ=SQRT((B**2)-4*A*C)
      DAT(10)=(-B+ROOTZ)/(2*A)
C       PRES(DBARS)=.980665*(1.02940Z+2.022365*10**-6*Z**2)   Z IN METERS
C
C       CALCULATE BRUNT-VAISALA FREQUENCY
C       AVERAGE DENSITIES WITHIN EACH BIN AND THEN COMPUTE BV
C
31    IF(IAX(1).NE.11.AND.IAX(2).NE.11.AND.IAX(3).NE.11) GO TO 756
C       CHECK FOR 1ST POINT OF NEW SEGMENT
      IF(NUMBV.NE.1) GOTO 33
      DMX2=DPTBV
      DMN2=DPTBV-AVDPT
      IF(DAT(10).GE.DMN2.AND.DAT(10).LE.DMX2) DFRST=DAT(10)
      IF(DAT(10).GE.DMN2.AND.DAT(10).LE.DMX2) GO TO 252
      DLAST=DAT(10)
      DPTBV=DLAST+AVDPT
      GO TO 20
C       SEE IF POINT WITHIN BOUNDS,  IFSO ADD TO AVERAGE
33    DMX1=DPTBV
      DMN1=DPTBV-AVDPT
      IF(DAT(10).GT.DMX1.OR.DAT(10).LT.DMN1)GO TO 250
```

```
         IF(NUMBV.GT.500)GO TO 250
C           DEPTHS TOO CLOSE TOGETHER CAUSES ARITH. PROBS IN
C           DIFFERENCE METHOD, SO DON'T ALLOW
         IF(DAT(10)-DLAST.LT.0.05)GO TO 20
252      XBV(NUMBV)=DENS
         DLAST=DAT(10)
         NUMBV=NUMBV+1
         GO TO 20
C          AVERAGE DENSITIES, COMPUTE BV, AND REINITIALIZE
250      SUMRH=0.0
C          DO NOT USE INTERVAL UNLESS HAS CONTAINED AT LEAST 4 VALID PTS
         IF(NUMBV.GT.5) GO TO 255
         NUMBV=1
         DPTBV=DLAST+AVDPT
         GO TO  20
255      DO 260 J=1,NUMBV-1
         SUMRH=SUMRH+XBV(J)
260      CONTINUE
         SUMRH=SUMRH/FLOAT(NUMBV-1)
         CALL BRVA1(DPTBV,SUMRH,RHO0,BV)
         NUMBV=1
         DPTBV=DLAST+AVDPT
         DAT(11)=BV
C          FOR PLOTTING PURPOSES PUT DAT(10) = AVG FOR INTERVAL
         DAT(10)=(DLAST+DFRST)/2.0
         WRITE(6,270)DAT(10),SUMRH,DAT(11)
270      FORMAT(1X,"DEPTH=",F10.2,4X,"RHO=",F10.6,4X,"BV FREQ=",F10.2)
C
C          TIME PLOTS
C
756      IF(IAX(1).NE.12.AND.IAX(3).NE.12.AND.IAX(2).NE.12) GO TO 84
         IF(NDAT1.GT.IBUF2(2))GO TO 15
         IF(NDAT1.EQ.IBUF2(2).AND.NDAT2.GT.IBUF2(3))GO TO 15
         NDEL1=IBUF2(2)-NDAT1
         NDEL2=IBUF2(3)-NDAT2+(NDEL1*3600)
         DAT(12)=(FLOAT(NDEL2)/60.)+((FLOAT(I-8))/5760.)
         IF(IAX(1).EQ.12)TX=X1AXL*X1INCR
         IF(IAX(3).EQ.12)TX=X2AXL*X2INCR
         IF(IAX(2).EQ.12)TX=YAXL*YINCR
         IF(DAT(12).GT.TX) GO TO 99
C
C          COMPUTE DEL(T)/DEL(P) AND DEL(S)/DEL(P)
C
84       IF(IAX(1).EQ.13.OR.IAX(2).EQ.13.OR.IAX(3).EQ.13)GO TO 85
         IF(IAX(1).EQ.14.OR.IAX(2).EQ.14.OR.IAX(3).EQ.14)GO TO 85
         GO TO 23
85       DTMP=DAT(3)-TOLD
         DSAL=DAT(1)-SOLD
         DPRS=DAT(2)-POLD
C        WRITE(6,977)DAT(3),DAT(1),DAT(2),DTMP,DSAL,DPRS
C977     FORMAT(1X,6(F12.4,2X))
         IF(ABS(DPRS).LT.0.001)GO TO 20
         DAT(13)=DTMP/DPRS
         DAT(14)=DSAL/DPRS
C        WRITE(6,978)DAT(13),DAT(14)
C978     FORMAT(1X,2(F12.4,2X))
         TOLD=DAT(3)
         SOLD=DAT(1)
         IF(POLD.NE.0.0)ITMP5=0
```

```
      IF(POLD.EQ.0.0)ITMP5=1
      POLD=DAT(2)
      IF(ITMP5.EQ.1)GO TO 20
C
C       PLOT OUT THE DATA
C
23    CALL PLT(DAT(IAX(1)),DAT(IAX(2)),DAT(IAX(3)),IOPT)
20    CONTINUE
      GO TO 15
99    ICOPY=15427B
      WRITE(7,754)ICOPY
754   FORMAT(A2)
      END
      SUBROUTINE PLT(X1,Y,X2,IOPT)
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON X1AXL,X1IV,X1INCR,YAXL,YIV,YINCR,X2AXL,X2IV,X2INCR
      X1MAX=X1IV+(X1INCR*X1AXL)
      YEXT=YIV+(YINCR*YAXL)
      YMAX=AMAX1(YEXT,YIV)
      YMIN=AMIN1(YEXT,YIV)
      X2MAX=X2IV+(X2INCR*X2AXL)
      IF(X1.LT.X1IV.OR.X1.GT.X1MAX)RETURN
      IF(IOPT.EQ.2) GO TO 118
      IF(X2.LT.X2IV.OR.X2.GT.X2MAX)RETURN
118   IF(Y.LT.YMIN.OR.Y.GT.YMAX)RETURN
      X1SCAL=X1AXL/(X1MAX-X1IV)
      X1PLT=X1SCAL*(X1-X1IV)
      X2SCAL=X2AXL/(X2MAX-X2IV)
      X2PLT=X2SCAL*(X2-X2IV)
      YSCAL=YAXL/(YMAX-YMIN)
      IF(YINCR.GE.0)YPLT=YSCAL*(YMAX-Y)
      IF(YINCR.LT.0)YPLT=YSCAL*(Y-YMIN)
      NUMB1=3
      NUMB2=1
      CALL SYMB(X1PLT,YPLT,.04,NUMB1,0.0,-1)
      IF(IOPT.EQ.2) GO TO 117
      CALL SYMB(X2PLT,YPLT,.04,NUMB2,0.0,-1)
117   RETURN
      END
      SUBROUTINE INIT1
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON X1AXL,X1IV,X1INCR,YAXL,YIV,YINCR,X2AXL,X2IV,X2INCR
      INSTR=1
      TCTP=0.0
      C1=0.1
      C2=0.0
      C3=0.0
      C4=.0005
      C5=0.0010
      C6=0.0
      INTCP1=4
      LAG=0
      WRITE(1,18)
```

```
18    FORMAT("USE DEFAULT VALUES, 1=YES 2=NO")
      READ(1,*)KANS
      IF(KANS.EQ.1)GO TO 99
      WRITE(1,20)
20    FORMAT("DEFAULT VALUES GIVEN IN PARENTHS")
      WRITE(1,22)
22    FORMAT("INSTRUMENT # (1)")
      READ(1,*)INSTR
      WRITE(1,24)
24    FORMAT("TEMP PROBE TIME CONST,SECONDS (0.0)")
      READ(1,*)TCTP
      WRITE(1,26)
26    FORMAT("CAL CONST PRESSURE C1,C2 (0.1,0.0)")
      READ(1,*)C1,C2
      WRITE(1,28)
28    FORMAT("CAL CONST TEMP C3,C4 (0.0,0.0005)")
      READ(1,*)C3,C4
      WRITE(1,30)
30    FORMAT("CAL CONST COND C5,C6 (0.001,0.0)")
      READ(1,*)C5,C6
      WRITE(1,40)
40    FORMAT("LAG OF TEMP PROBE #OF SCANS (0)")
      READ(1,*)LAG
C       WRITE OUT VALUES
      WRITE(1,64)INSTR,TCTP
64    FORMAT(1X,"INSTR ",I2,"TCTP ",F8.4)
      WRITE(1,66)C1,C2,C3,C4,C5,C6
66    FORMAT(1X,"CAL CONSTS ",6F8.4)
      WRITE(1,68)X1AXL,X1IV,X1INCR,YAXL,YIV,YINCR,X2AXL,X2IV,X2INCR
68    FORMAT(1X,"AXES CONSTS ",9F6.2)
      WRITE(1,70)INTCP1,LAG
70    FORMAT(1X,"PLT TM INT ",I4,"LAG IN SCANS ",I4)
99    RETURN
      END
      END$
```

REFERENCES

1. N.L. Brown and G.K. Morrison, "W.H.O.I/Brown Conductivity, Temperature and Depth Microprofiler," Woods Hole Oceanographic Institution Technical Report 78-23 (unpublished manuscript), 1978.

2. Neil L. Brown, "A Precision CTD Microprofiler," in *Ocean '74: IEEE International Conference on Engineering in the Ocean Environment Record,* Vol. 2, p. 270, 1974.

3. N.P. Fofonoff, S.P. Hayes, and R.C. Millard Jr., "W.H.O.I./ Brown CTD Microprofiler: Methods of Calibration and Data Handling," Woods Hole Oceanographic Institution Technical Report 74-89 (unpublished manuscript), 1974.

4. A. Bradshaw, and K.E. Schleicher, "The Effect of Pressure on the Electrical Conductance of Sea Water," Deep-Sea Res. **12**, 151-162 (1965).

5. N. Brown and B. Allentoft, "Salinity, Conductivity and Temperature Relationships of Sea Water over the Range of 0 to 60 p.p.t.," Bissett-Berman Corp. manuscript report, Mar. 1, 1966.

Appendix
SCIENTIFIC SUBROUTINES—FORTRAN SOURCE LISTINGS

```
FTN4,L
      SUBROUTINE LRSAL(G,P,T,S)
C       CTD PACKAGE SUBROUTINE TO OBTAIN SALINITY FROM C,T,P
C         MODIFIED TO FIT INTO MY PLOT PACKAGE. 18JULY77 L. ROSENBLUM
      COMMON IREC,ISTOP,ISCAN,NSCAN,IFLOR
      COMMON IBUF1(456),IBUF2(456)
      COMMON C1,C2,C3,C4,C5,C6,INSTR
      COMMON ISHIP(6),IST,IYR,IMO,IDAY,LOW
      COMMON PAXL,PIV,PINCR,TAXL,TIV,TINCR,SAXL,SIV,SINCR
C DEG CELSIUS; PRESSURE IN DECIBARS
      IF(ABS(S-35.0)-35.0) 3,3,2
    2 S=35.0
C  SCALE CTD TEMPERATURE TO THE 1948 STANDARD
    3 CONTINUE
      XT=T+4.4E-6*T*(100.-T)
      XC=G
      XP=P
      XTO = XT-15.
      RC=XC/42.909
      AL = -7.6E-5*XT+6.95E-3
      AJ = ((-1.657E-4*XT+8.276E-3)*XT-1.535E-1)*XT+1.0
      RT = (((-.67249142E-8*XT-.19426015E-6)*XT+.99886585E-4)*XT
     1+.20131661E-1)*XT+.67652453
      GA= ((-7.9E-6*XT+8.3089E-4)*XT-4.5302E-2)*XT+1.5192
      H=(-2.492E-9*XP+2.577E-5)*XP+4.0E-4
      F=((3.3E-13*XP-3.3913E-8)*XP+1.042E-3)*XP
      SP =GA*F +H*AJ
      RCT=RC/RT
      RP0 = 1. +1.0E-2*SP
      RP1=1.0E-2*SP*AL
    5 SO=S
      RP=RP0+RP1*(35.0-S)
      RS=RCT/RP
      R15 = ((-.21E-5*RS*RS-.63E-5)*XTO+(37.3E-5*RS-72.E-5)*RS
     1+96.7E-5)*XTO*RS*(RS-1.0) + RS
      S=((((-1.32311*R15+5.98624)*R15-10.67869)*R15+12.80832)
     1*R15+28.2972)*R15-0.08996
      IF(ABS(S-SO)-5E-2) 10,10,5
   10 CONTINUE
      END
      END$
```

```
FTN4,L
      SUBROUTINE POTMP(PRS,TEMP,SAL,PTMP)
C POTENTIAL TEMPERATURE ACCORDING TO BRYDEN
C CONVERT TO 1948 TEMPERATURE SCALE
      TMP = TEMP + 4.4E-6*TEMP*(100.-TEMP)
      S0=SAL-35.
      A3=.50484E-14*TMP-0.16056E-12
      A2=(.21987E-11*TMP-0.31628E-9)*TMP+.89309E-8
     1-.41057E-10*S0
      A11=-.29778E-7*TMP+.17439E-5
      A10=((.40274E-9*TMP-.54065E-7)*TMP
     1+.83198E-5)*TMP+.36504E-4
      A1=A11*S0+A10
      PTMP=TEMP-((A3*PRS+A2)*PRS+A1)*PRS
      RETURN
      END
```

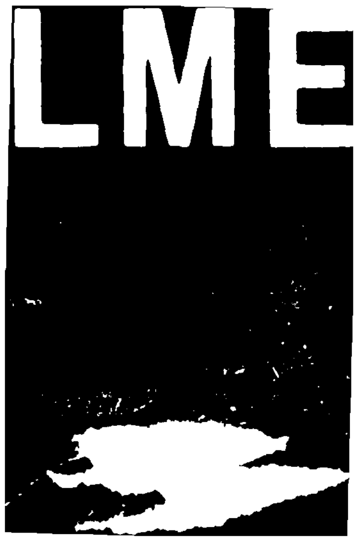
```
FTN4,L
      SUBROUTINE SVEL(P,T,S,V)
C     J WEBSTER           SVEL SUBROUTINE
C     COMPUTES VELOCITY OF SOUND IN METERS/SEC FROM PRESSURE IN DECIBARS
C     TEMPERATURES IN DEGREES C AND SALINITY IN PARTS PER THOUSAND
C     CHANGE PRESSURE TO TOTAL PRESSURE IN KG/CM**2
      PKC = P * .1019716 + 1.03323
      SM = S - 35.0
      C00 = (((7.9851E-6*T - 2.6045E-4)*T - 4.4532E-2)*T + 4.5721  )*T
     X    + 1449.14
      C01 = (7.7711E-7*T - 1.1244E-2)*T + 1.39799
      C10 = ((4.5283E-8*T + 7.4812E-6)*T - 1.8607E-4)*T + 1.60272E-1
      C11 = (1.5790E-9*T + 3.1580E-8)*T + 7.7016E-5
      C20 = (1.8563E-9*T - 2.5294E-7)*T + 1.0268E-5
      B3 = -1.9646E-10*T + 3.5216E-9
      B0 = (1.69202E-3*SM + C01)*SM + C00
      B1 = C11*SM + C10
      B2 = -1.2943E-7*SM + C20
      V = (((-3.3603E-12 * PKC + B3)*PKC+B2)*PKC+B1)*PKC+B0
      RETURN
      END
      END$
```

```
FTN4,L
C
      SUBROUTINE ATG(PRS,TEMP,SAL,GAMMA)
C COMPUTES ADIABATIC TEMPERATURE GRADIENT IN DEG CELSIUS/1000 DB
C CONVERT TO 1948 TEMPERATURE SCALE
      TMP = TEMP + 4.4E-6*TEMP*(100.-TEMP)
      S = SAL - 35.
      A01 = -.42393E-4*TMP+.18932E-2
      A00 = ((.66228E-6*TMP-.6836E-4)*TMP+.85258E-2)*TMP+.35803E-1
      A10 = ((-.54481E-10*TMP+.8733E-8)*TMP-.67795E-6)*TMP+.18741E-4
      A11 = .27759E-8*TMP-.11351E-6
      A2 = (-.21687E-12*TMP+.18676E-10)*TMP-.46206E-9
      A1 = A11*S + A10
      A0 = A01*S + A00
      GAMMA = (A2*PRS+A1)*PRS + A0
      RETURN
      END
      END$
```

```
FTN4,L
C
      SUBROUTINE SIGMT(TMP,XS,XS0,XST)
      DOUBLE PRECISION XT,XS0,E1,B1,B2,E2,XST
C CONVERT TO 1948 TEMPERATURE SCALE
      XT = TMP + 4.4D-6*TMP*(100.-TMP)
      XS0 = ((6.76786136D-6*XS-4.82496140D-4)*XS+8.14876577D-1)*XS
     X   -9.34458632D-2
      E1 = (((-1.43803061D-7*XT-1.98248399D-3)*XT-5.45939111D-1)*XT
     X    +4.53168426D0)*XT
      B1 = ((-1.0843D-6*XT+9.8185D-5)*XT-4.7867D-3)*XT + 1.0
      B2 = ((1.667D-8*XT-8.164D-7)*XT + 1.803D-5)*XT
      E2=(B2   *XS0+B1) * XS0
      XST = E1/(XT+6.726D1  )+E2
      RETURN
      END
      END$
```

```
FTN4,L
C
      SUBROUTINE SPVOL(PRS,TEMP,XSO,XST,XALPH)
      DOUBLE PRECISION A0,E2,XALPH,XSO,XST,TMP,B10,B11,B12,B20,B21,
     1                 B22,C3,C1,C2,E1
C
C SPECIFIC VOLUME IN CM**3/GM
C
C CONVERT TO 1948 TEMPERATURE SCALE
      TMP = TEMP + 4.4E-6*TEMP*(100.-TEMP)
      B10=((4.0E-12*TMP-6.63E-10)*TMP+3.673E-8)*TMP-2.2072E-7
      B11=(4.0E-12*TMP-3.28E-10)*TMP+1.725E-8
      B12=1.0E-12*TMP-4.5E-11
      B20=(2.14E-14*TMP-1.24064E-12)*TMP-6.68E-14
      B21=(-2.0E-16*TMP+1.206E-14)*TMP-4.248E-13
      B22=-6.0E-17*TMP+1.8E-15
      C3=1.5E-17*TMP
      C1=(B12*XSO+B11)*XSO+B10
      C2=(B22*XSO+B21)*XSO+B20
      E1=((C3*PRS+C2)*PRS+C1)*PRS
      E2=1.0-4.886E-6*PRS/(1.0+1.83E-5*PRS)+E1
      A0=1.0/(1.0+.001*XST)
      XALPH=A0*E2
      RETURN
      END
      END$
```

```
FTN4,L
      SUBROUTINE BRVA1(DPT,RHO,RHO0,BV)
      DOUBLE PRECISION RDIF,E,RHO,RHOL,DDIF,GSPVO
C       SUBROUTINE COMPUTES BRUNT-VAISALA FREQ. IN CYCLES/HR
C       USING APPROX. GIVEN IN PHILLIPS P. 17
C
C       PROGRAMMER L. ROSENBLUM 10/78
C
C       CONST IS GRAVITY IN M/SEC/SEC
      GSPVO=9.81/RHO0
      DDIF=DPT-DPTL
      RDIF=RHO-RHOL
      E=GSPVO*RDIF/DDIF
C       CONST CONVERTS FROM RADIANS/SEC TO CYCLES/HR
      BVV=572.9578*DSQRT(DABS(E))
      BV=SIGN(BVV,E)
      RHOL=RHO
      DPTL=DPT
      RETURN
      END
      END$
```


LME